*3GPP2 C.S0024*

*Version 2.0*

*Date: October 27, 2000*

3RD GENERATION
PARTNERSHIP
PROJECT 2
"3GPP2"

# *cdma2000 High Rate Packet Data Air Interface Specification*

**CONTENTS**

CONTENTS

**CONTENTS**

**CONTENTS**

**CONTENTS**

**CONTENTS**

**CONTENTS**

**CONTENTS**

CONTENTS

**CONTENTS**

CONTENTS

CONTENTS

CONTENTS

**CONTENTS**

**CONTENTS**

CONTENTS

**CONTENTS**

**CONTENTS**

CONTENTS

CONTENTS

**CONTENTS**

**CONTENTS**

**CONTENTS**

**CONTENTS**

**CONTENTS**

CONTENTS

**CONTENTS**

CONTENTS

**CONTENTS**

**CONTENTS**

## FIGURES

**FIGURES**

**FIGURES**

## FIGURES

**TABLES**

**TABLES**

**TABLES**

**TABLES**

# FOREWORD

## (This foreword is not part of this Standard)

This standard was prepared by Technical Specification Group C of the Third Generation Partnership Project 2 (3GPP2). This standard is evolved from and is a companion to the cdma2000 standards. This air interface standard provides high rate packet data services.

Ten different operating bands have been specified. Equipment built to this standard can be used in a band subject to the allocation of the band and to the rules and regulations of the country to which the allocated band has been assigned.

## REFERENCES

The following standards contain provisions, which, through reference in this text, constitute provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below.

[1] P.S0001-A, Wireless IP Network Standard.

[2] C.S0002-A, Physical Layer Standard for cdma2000 Spread Spectrum Systems.

[3] C.S0005-A, Upper Layer (Layer 3) Signaling Specification for cdma2000 Spread Spectrum Systems.

[4] C.P9011, Recommended Minimum Performance Standards for cdma2000 High Rate Packet Data Access Network.

[5] C.P9012, Recommended Minimum Performance Standards for cdma2000 High Rate Packet Data Access Terminal.

[6] FIPS PUB 180-1, Federal Information Processing Standards Publication 180-1.

[7] RFC 2409, The Internet Key Exchange (IKE).

[8] RFC 1700, Assigned Numbers.

[9] A.S0001, 3GPP2 Access Network Interfaces Interoperability Specification.

# 1 OVERVIEW

## 1.1 Scope of This Document

These technical requirements form a compatibility standard for cdma2000 high rate packet data systems. These requirements ensure that a compliant access terminal can obtain service through any access network conforming to this standard. These requirements do not address the quality or reliability of that service, nor do they cover equipment performance or measurement procedures.

This specification is primarily oriented toward requirements necessary for the design and implementation of access terminals. As a result, detailed procedures are specified for access terminals to ensure a uniform response to all access networks. Access network procedures, however, are specified only to the extent necessary for compatibility with those specified for the access terminal.

This specification includes provisions for future service additions and expansion of system capabilities. The architecture defined by this specification permits such expansion without the loss of backward compatibility to older access terminals.

This compatibility standard is based upon spectrum allocations that have been defined by various governmental administrations. Those wishing to deploy systems compliant with this standard should also take notice of the requirement to be compliant with the applicable rules and regulations of local administrations. Those wishing to deploy systems compliant with this standard should also take notice of the electromagnetic exposure criteria for the general public and for radio frequency carriers with low frequency amplitude modulation.

## 1.2 Requirements Language

Compatibility, as used in connection with this standard, is understood to mean: Any access terminal can obtain service through any access network conforming to this standard. Conversely, all access networks conforming to this standard can service access terminals.

"Shall" and "shall not" identify requirements to be followed strictly to conform to the standard and from which no deviation is permitted. "Should" and "should not" indicate that one of several possibilities is recommended as particularly suitable, without mentioning or excluding others, that a certain course of action is preferred but not necessarily required, or that (in the negative form) a certain possibility or course of action is discouraged but not prohibited. "May" and "need not" indicate a course of action permissible within the limits of the standard. "Can" and "cannot" are used for statements of possibility and capability, whether material, physical, or causal.

## 1.3 Architecture Reference Model

The architecture reference model is presented in Figure 1.3-1. The reference model consists of the following functional units:

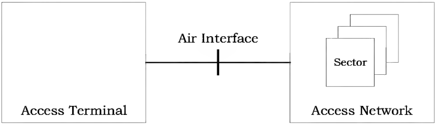


**Figure 1.3-1. Architecture Reference Model**

The access terminal, the access network, and the sector are formally defined in 1.11.

The reference model includes the air interface between the access terminal and the access network. The protocols used over the air interface are defined in this document.

## 1.4 Protocol Architecture

The air interface has been layered, with interfaces defined for each layer (and for each protocol within each layer). This allows future modifications to a layer or to a protocol to be isolated.

### 1.4.1 Layers

Figure 1.4.1-1 describes the layering architecture for the air interface. Each layer consists of one or more protocols that perform the layer's functionality. Each of these protocols can be individually negotiated.

| |
|---|
| Application Layer |
| Stream Layer |
| Session Layer |
| Connection Layer |
| Security Layer |
| MAC Layer |
| Physical Layer |

**Figure 1.4.1-1. Air Interface Layering Architecture**

The protocols and layers specified in Figure 1.4.1-1 are:

1. Application Layer: The Application Layer provides multiple applications. It provides the Default Signaling Application for transporting air interface protocol messages. The Default Signaling Application is defined in Chapter 2. It also provides the Default Packet Application for transporting user data. The Default Packet Application is defined in Chapter 3.

2. Stream Layer: The Stream Layer provides multiplexing of distinct application streams. Stream 0 is dedicated to signaling and defaults to the Default Signaling Application (see Chapter 2). Stream 1, Stream 2, and Stream 3 are not used by default. The Stream Layer is defined in Chapter 4.

3. Session Layer: The Session Layer provides address management, protocol negotiation, protocol configuration and state maintenance services. The Session Layer is defined in Chapter 5.

4. Connection Layer: The Connection Layer provides air link connection establishment and maintenance services. The Connection Layer is defined in Chapter 6.

5. Security Layer: The Security Layer provides authentication and encryption services. The Security Layer is defined in Chapter 7.

6. MAC Layer: The Medium Access Control (MAC) Layer defines the procedures used to receive and to transmit over the Physical Layer. The MAC Layer is defined in Chapter 8.

7. Physical Layer: The Physical Layer provides the channel structure, frequency, power output, modulation, and encoding specifications for the Forward and Reverse Channels. The Physical Layer is defined in Chapter 9.

Each layer may contain one or more protocols. Protocols use signaling messages or headers to convey information to their peer entity at the other side of the air-link. When protocols send messages they use the Signaling Network Protocol (SNP) to transmit these messages.

## 1.5 Physical Layer Channels

The Physical Layer defines the Physical Layer Channels and the Forward and Reverse Channel hierarchies shown in Figure 1.5-1 and Figure 1.5-2. Channel *x* is part of Channel *y* if *y* is an ancestor of *x*. The specific channels are defined in 1.11. When the context is clear, the complete qualified name is usually omitted (e.g., Pilot Channel as opposed to Forward Pilot Channel or Data Channel as opposed to Reverse Traffic Data Channel).

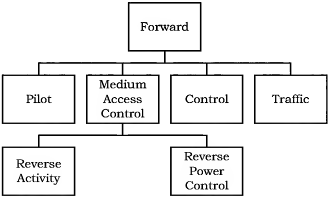


**Figure 1.5-1. Forward Channel Structure**

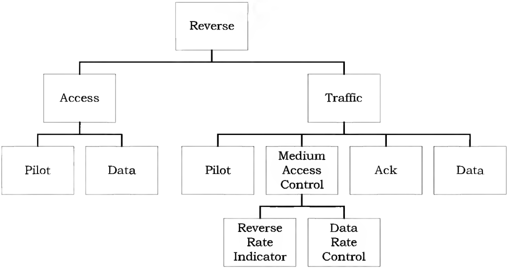


**Figure 1.5-2. Reverse Channel Structure**

## 1.6 Protocols

### 1.6.1 Interfaces

This standard defines a set of interfaces for communications between protocols in the same entity and between a protocol executing in one entity and the same protocol executing in the other entity.

In the following the generic term "entity" is used to refer to the access terminal and the access network.

Protocols in this specification have four types of interfaces:

- Headers and messages are used for communications between a protocol executing in one entity and the same protocol executing in the other entity.
- Commands are used by a higher layer protocol to obtain a service from a lower layer protocol in the same entity. Commands can be sent between protocols in the same layer but only in one direction (i.e., if protocol A and protocol B are in the same layer and protocol A sends a command to protocol B, protocol B cannot send a command to protocol A). For example, *AccessChannelMAC.Abort* causes the Access Channel MAC Protocol to abort any access attempt currently in progress.

- Indications are used by a lower layer protocol to convey information regarding the occurrence of an event. Any higher layer protocol can register to receive these indications. A same layer protocol can also register to receive an indication but only in one direction (if protocol A and protocol B are in the same layer and protocol A registers to receive an indication from protocol B, protocol B cannot register to receive an indication from protocol A.). For example, the access terminal Reverse Traffic Channel MAC Protocol returns a "Reverse Link Acquired" indication when it gets a message from its peer protocol at the access network that it has acquired the Reverse Traffic Channel. This notification is then used by Connection Layer protocols to continue with the handshake leading to the establishment of the connection.
- Public Data is used to share information in a controlled way between protocols. Public data is shared between protocols in the same layer, as well as between protocols in different layers. An example of this is the MinimumProtocolRevision made public by the Connection Layer Initialization State Protocol after the protocol receives it in the Sync message.

Commands and indications are written in the form of *Protocol.Command* and *Protocol.Indication*. For example, *AccessChannelMAC.Activate* is a command activating the Access Channel MAC, and *IdleState.ConnectionOpened* is an indication provided by the Connection Layer Idle State Protocol that the connection is now open. When the context is clear, the *Protocol* part is dropped (e.g., within the Idle State Protocol, *Activate* refers to *IdleState.Activate*).

Commands are always written in the imperative form, since they direct an action. Indications are always written in the past tense since they notify of events that happened (e.g., *OpenConnection* for a command and *ConnectionOpened* for an indication).

Headers and messages are binding on all implementations. Commands, indications, and public data are used as a device for a clear and precise specification. Access terminals and access networks can be compliant with this specification while choosing a different implementation that exhibits identical behavior.

### 1.6.2 States

When protocols exhibit different behavior as a function of the environment (e.g., if a connection is opened or not, if a session is opened or not, etc.), this behavior is captured in a set of states and the events leading to a transition between states.

Unless otherwise specifically mentioned, the state of the access network refers to the state of a protocol engine in the access network as it applies to a particular access terminal. Since the access network communicates with multiple access terminals, multiple independent instantiations of a protocol will exist in the access network, each with its own independent state machine.

Typical events leading to a transition from one state to another are the receipt of a message, a command from a higher layer protocol, an indication from a lower layer protocol, or the expiration of a timer.

When a protocol is not functional at a particular time (e.g., the Access Channel MAC protocol at the access terminal when the access terminal has an open connection) the protocol is placed in a state called the Inactive state. This state is common for most protocols.

Other common states are Open, indicating that the session or connection (as applicable to the protocol) is open and Close, indicating that the session or connection is closed.

If a protocol has a single state other than the Inactive state, that state is always called the Active state. If a protocol has more than one state other than the Inactive state, all of these states are considered active, and are given individual names (e.g., the Forward Traffic Channel MAC protocol has three states: Inactive, Variable Rate, and Fixed Rate).

### 1.6.3 Common Commands

Most protocols support the following two commands:

- *Activate,* which commands the protocol to transition away from the Inactive state to some other state.
- *Deactivate,* which commands the protocol to transition to the Inactive state. Some protocols do not transition immediately to the Inactive state, due to requirements on orderly cleanup procedures.

Other common commands are *Open* and *Close,* which command protocols to perform session open /close or connection open / close related functions.

### 1.6.4 Protocol Negotiation

Most protocols can be negotiated and can be configured when the session is set-up (see 1.9 for a discussion of sessions). Protocols are associated with a Type that denotes the type of the protocol (e.g., Access Channel MAC Protocol) and with a Subtype that denotes a specific instance of a protocol (e.g., the Default Access Channel MAC Protocol and perhaps one day, the Extended and Bloated Access Channel MAC Protocol).

The negotiation and configuration processes are part of the Session Layer.

### 1.6.5 Protocol Overview

Figure 1.6.5-1 presents the default protocols defined for each one of the layers shown in Figure 1.4.1-1. The following is a brief description of each protocol. A more complete description is provided in the Introduction section of each layer.

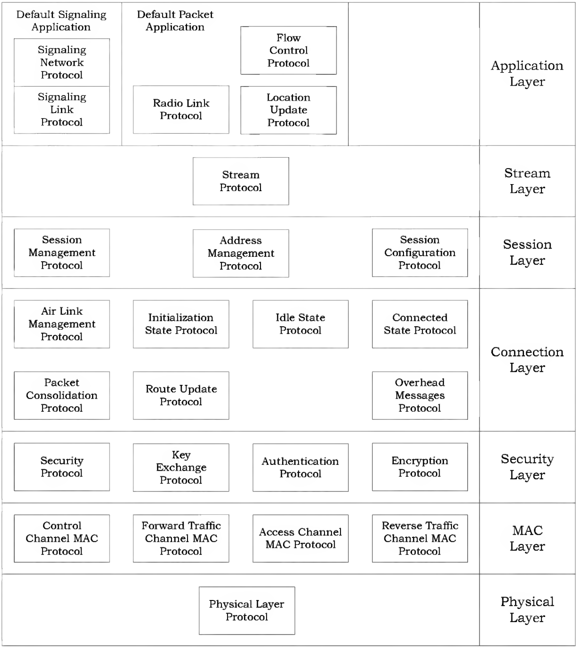

Default Signaling Application
Signaling Network Protocol
Signaling Link Protocol
Default Packet Application
Flow Control Protocol
Radio Link Protocol
Location Update Protocol
Application Layer
Stream Protocol
Stream Layer
Session Management Protocol
Address Management Protocol
Session Configuration Protocol
Session Layer
Air Link Management Protocol
Initialization State Protocol
Idle State Protocol
Connected State Protocol
Packet Consolidation Protocol
Route Update Protocol
Overhead Messages Protocol
Connection Layer
Security Protocol
Key Exchange Protocol
Authentication Protocol
Encryption Protocol
Security Layer
Control Channel MAC Protocol
Forward Traffic Channel MAC Protocol
Access Channel MAC Protocol
Reverse Traffic Channel MAC Protocol
MAC Layer
Physical Layer Protocol
Physical Layer

**Figure 1.6.5-1. Default Protocols**

- Application Layer:
  - Default Signaling Application:
    - Signaling Network Protocol: The Signaling Network Protocol (SNP) provides message transmission services for signaling messages.
    - Signaling Link Protocol: The Signaling Link Protocol (SLP) provides fragmentation mechanisms, along with reliable and best-effort delivery mechanisms for signaling messages. When used in the context of the Default Signaling Application, SLP carries SNP packets.
  - Default Packet Application:
    - Radio Link Protocol: The Radio Link Protocol (RLP) provides retransmission and duplicate detection for an octet aligned data stream.
    - Location Update Protocol: The Location Update Protocol defines location update procedures and messages in support of mobility management for the Default Packet Application.
    - Flow Control Protocol: The Flow Control Protocol defines flow control procedures to enabling and disabling the Default Packet Application data flow.
- Stream Layer:
  - Stream Protocol: adds the stream header in the transmit direction; removes the stream header and forwards packets to the correct application on the receiving entity.
- Session Layer:
  - Session Management Protocol: provides means to control the activation and the deactivation of the Address Management Protocol and the Session Configuration Protocol. It also provides a session keep alive mechanism.
  - Address Management Protocol: Provides access terminal identifier (ATI) management.
  - Session Configuration Protocol: Provides negotiation and configuration of the protocols used in the session.
- Connection Layer:
  - Air Link Management Protocol: Provides the overall state machine management that an access terminal and an access network follow during a connection.
  - Initialization State Protocol: Provides the procedures that an access terminal follows to acquire a network and that an access network follows to support network acquisition.
  - Idle State Protocol: Provides the procedures that an access terminal and an access network follow when a connection is not open.
  - Connected State Protocol: Provides the procedures that an access terminal and an access network follow when a connection is open.

  - Route Update Protocol: Provides the means to maintain the route between the access terminal and the access network.
  - Overhead Messages Protocol: Provides broadcast messages containing information that is mostly used by Connection Layer protocols.
  - Packet_Consolidation_Protocol: Provides transmit prioritization and packet encapsulation for the Connection Layer.
- Security Layer:
  - Key Exchange Protocol: Provides the procedures followed by the access network and the access terminal to exchange security keys for authentication and encryption.
  - Authentication Protocol: Provides the procedures followed by the access network and the access terminal for authenticating traffic.
  - Encryption Protocol: Provides the procedures followed by the access network and the access terminal for encrypting traffic.
  - Security Protocol: Provides procedures for generation of a cryptosync that can be used by the Authentication Protocol and Encryption Protocol.
- MAC Layer:
  - Control Channel MAC Protocol: Provides the procedures followed by the access network to transmit, and by the access terminal to receive the Control Channel.
  - Access Channel MAC Protocol: Provides the procedures followed by the access terminal to transmit, and by the access network to receive the Access Channel.
  - Forward Traffic Channel MAC Protocol: Provides the procedures followed by the access network to transmit, and by the access terminal to receive the Forward Traffic Channel.
  - Reverse Traffic Channel MAC Protocol: Provides the procedures followed by the access terminal to transmit, and by the access network to receive the Reverse Traffic Channel.
- Physical Layer:
  - Physical Layer Protocol: Provides channel structure, frequency, power output and modulation specifications for the forward and reverse links.

## 1.7 Default Applications

This document defines two default applications that all compliant access terminals and access networks support:

- Default Signaling Application, which provides the means to carry messages between a protocol in one entity and the same protocol in the other entity. The Default Signaling Application consists of a messaging protocol (Signaling Network Protocol) and a link layer protocol that provides message fragmentation, retransmission and duplicate detection (Signaling Link Protocol).

- Default Packet Application. The Default Packet Application consists of a link layer protocol that provides octet retransmission and duplicate detection (Radio Link Protocol), a location update protocol that provides mobility between data service networks and a flow control protocol that provides flow control of data traffic.

The applications used and the streams upon which they operate are negotiated as part of session negotiation.

## 1.8 Streams

The air interface can support up to four parallel application streams. The first stream (Stream 0) always carries Signaling, and the other three can be used to carry applications with different Quality of Service (QoS) requirements or other applications.

## 1.9 Sessions and Connections

A session refers to a shared state between the access terminal and the access network. This shared state stores the protocols and protocol configurations that were negotiated and are used for communications between the access terminal and the access network.

Other than to open a session, an access terminal cannot communicate with an access network without having an open session.

A connection is a particular state of the air-link in which the access terminal is assigned a Forward Traffic Channel, a Reverse Traffic Channel and associated MAC Channels.

During a single session the access terminal and the access network can open and can close a connection multiple times.

## 1.10 Security

The air interface supports a security layer, which can be used for authentication and encryption of access terminal traffic transported by the Control Channel, the Access Channel, the Forward Traffic Channel and the Reverse Traffic Channel.

## 1.11 Terms

**Access Network (AN).** The network equipment providing data connectivity between a packet switched data network (typically the Internet) and the access terminals. An access network is equivalent to a base station in [2].

**Access Terminal (AT).** A device providing data connectivity to a user. An access terminal may be connected to a computing device such as a laptop personal computer or it may be a self-contained data device such as a personal digital assistant. An access terminal is equivalent to a mobile station in [2].

**ATI**. Access Terminal Identifier.

**BATI**. Broadcast Access Terminal Identifier.

**CDMA System Time in Slots**. An integer value $s$ such that: $s = \lfloor t \times 600 \rfloor$, where $t$ represents CDMA System Time in seconds. Whenever the document refers to the CDMA System Time in slots, it is referring to the value $s$.

**CDMA System Time**. The time reference used by the system. CDMA System Time is synchronous to UTC time except for leap seconds and uses the same time origin as GPS time. Access terminals use the same CDMA System Time, offset by the propagation delay from the access network to the access terminal.

**Channel**. The set of channels transmitted between the access network and the access terminals within a given frequency assignment. A Channel consists of a Forward Link and a Reverse Link.

**Connection Layer**. The Connection Layer provides air link connection establishment and maintenance services. The Connection Layer is defined in Chapter 6.

**Dedicated Resource**. An access network resource required to provide any data service to the access terminal, e.g, Wireless IP Service (see [1]) that is granted to the access terminal only after access terminal authentication has completed successfully. Power control and rate control are not considered dedicated resources.

**Forward Channel**. The portion of the Channel consisting of those Physical Layer Channels transmitted from the access network to the access terminal.

**Forward Control Channel**. The channel that carries data to be received by all access terminals monitoring the Forward Channel.

**Forward MAC Channel**. The portion of the Forward Channel dedicated to Medium Access Control activities. The Forward MAC Channel consists of the RPC, and RA Channels.

**Forward MAC Reverse Activity (RA) Channel**. The portion of the Forward MAC Channel that indicates activity level on the Reverse Channel.

**Forward MAC Reverse Power Control (RPC) Channel**. The portion of the Forward MAC Channel that controls the power of the Reverse Channel for one particular access terminal.

**Forward Pilot Channel**. The portion of the Forward Channel that carries the pilot.

**Forward Traffic Channel**. The portion of the Forward Channel that carries information for a specific access terminal. The Forward Traffic Channel can be used as either a Dedicated Resource or a non-Dedicated Resource. Prior to successful access terminal authentication, the Forward Traffic Channel serves as a non-Dedicated Resource. Only after successful access terminal authentication can the Forward Traffic Channel be used as a Dedicated Resource for the specific access terminal.

**Frame**. The duration of time specified by 16 slots or 26.66... ms.

**Global Positioning System (GPS)**. A US government satellite system that provides location and time information to users. See Navstar GPS Space Segment/Navigation User Interfaces ICD-GPS-200 for specifications

**$I_{BATI}$**. Index to the ReceiveATIList identifying the ReceiveATIList structure corresponding to BATI.

**$I_{currentUATI}$**. Index to the ReceiveATIList identifying the ReceiveATIList structure corresponding to the current ATI.

**$I_{newUATI}$**. Index to the ReceiveATIList identifying the ReceiveATIList structure corresponding to newly received ATI.

**$I_{RATI}$**. Index to the ReceiveATIList identifying the ReceiveATIList structure corresponding to RATI.

**MAC Layer**. The MAC Layer defines the procedures used to receive and to transmit over the Physical Layer. The MAC Layer is defined in Chapter 8.

**MATI**. Multicast Access Terminal Identifier.

**NULL.** A value which is not in the specified range of the field.

**Physical Layer**. The Physical Layer provides the channel structure, frequency, power output, modulation, and encoding specifications for the forward and reverse links. The Physical Layer is defined in Chapter 9.

**RATI**. Random Access Terminal Identifier.

**Reverse Access Channel.** The portion of the Reverse Channel that is used by access terminals to communicate with the access network when they do not have a traffic channel assigned. There is a separate Reverse Access Channel for each sector of the access network.

**Reverse Access Data Channel**. The portion of the Access Channel that carries data.

**Reverse Access Pilot Channel.** The portion of the Access Channel that carries the pilot.

**Reverse Channel**. The portion of the Channel consisting of those Physical Layer Channels transmitted from the access terminal to the access network.

**Reverse Traffic Ack Channel**. The portion of the Reverse Traffic Channel that indicates the success or failure of the Forward Traffic Channel reception.

**Reverse Traffic Channel.** The portion of the Reverse Channel that carries information from a specific access terminal to the access network. The Reverse Traffic Channel can be used as either a Dedicated Resource or a non-Dedicated Resource. Prior to successful access terminal authentication, the Reverse Traffic Channel serves as a non-Dedicated Resource. Only after successful access terminal authentication can the Reverse Traffic Channel be used as a Dedicated Resource for the specific access terminal.

**Reverse Traffic Data Channel.** The portion of the Reverse Traffic Channel that carries user data.

**Reverse Traffic MAC Channel**. The portion of the Reverse Traffic Channel dedicated to Medium Access Control activities. The Reverse Traffic MAC Channel consists of the RRI and DRC Channels.

**Reverse Traffic MAC Data Rate Control (DRC) Channel**. The portion of the Reverse Traffic Channel that indicates the rate at which the access terminal can receive the Forward Traffic Channel.

**Reverse Traffic MAC Reverse Rate Indicator (RRI) Channel**. The portion of the Reverse Traffic Channel that indicates the rate of the Reverse Traffic Data Channel.

**Reverse Traffic Pilot Channel.** The portion of the Reverse Traffic Channel that carries the pilot.

**RLP**. Radio Link Protocol provides retransmission and duplicate detection for an octet-aligned data stream.

**Sector.** The part of the access network that provides one CDMA channel.

**Security Layer**. The Security Layer provides authentication and encryption services. The Security Layer is defined in Chapter 7.

**Session Layer**. The Session Layer provides protocol negotiation, protocol configuration, and state maintenance services. The Session Layer is defined in Chapter 5.

**Slot**. A duration of time specified by 1.66... ms.

**SLP**. Signaling Link Protocol provides best-effort and reliable-delivery mechanisms for signaling messages. SLP is defined in 2.4.

**SNP**. Signaling Network Protocol provides message transmission services for signaling messages. The protocols that control each layer use SNP to deliver their messages to their peer protocols.

**Stream Layer**. The Stream Layer provides multiplexing of distinct streams. Stream 0 is dedicated to signaling and defaults to the default signaling stream (SNP / SLP) and Stream 1 defaults to the default packet service (RLP). Stream 2 and Stream 3 are not used by default. The Stream Layer is defined in Chapter 4.

**Subnet Mask (of length *n*).** A 128-bit value whose binary representation consists of *n* consecutive '1's followed by 128-*n* consecutive '0's.

**UATI**. Unicast Access Terminal Identifier.

**Universal Coordinated Time (UTC)**. An internationally agreed-upon time scale maintained by the Bureau International de l'Heure (BIH) used as the time reference by nearly all commonly available time and frequency distribution systems.

**UTC**. Universal Temps Coordine. See Universal Coordinated Time.

## 1.12 Notation

**A[i]** The ith element of array A. The first element of the array is A[0].

**<$e_1$, $e_2$, ..., $e_n$>** A *structure* with elements 'e1', 'e2', ..., 'en'.
Two structures E = <$e_1$, $e_2$, ..., $e_n$> and F = <$f_1$, $f_2$, ..., $f_m$> are equal iff 'm' is equal to 'n' and $e_i$ is equal to $f_i$ for i=1, ...n.
Given E = <$e_1$, $e_2$, ..., $e_n$> and F = <$f_1$, $f_2$, ..., $f_m$>, the assignment "E = F" denotes the following set of assignments: $e_i$ = $f_i$, for i=1, ...n.

**S.e** The member of the structure 'S' that is identified by 'e'.

**M[i:j]** Bits ith through jth inclusive (i ≥ j) of the binary representation of variable M. M[0:0] denotes the least significant bit of M.

| | |
|---|---|
| $\|$ | Concatenation operator. (A \| B) denotes variable A concatenated with variable B. |
| $\times$ | Indicates multiplication. |
| $\lfloor x \rfloor$ | Indicates the largest integer less than or equal to x: $\lfloor 1.1 \rfloor = 1$, $\lfloor 1.0 \rfloor = 1$. |
| $\lceil x \rceil$ | Indicates the smallest integer greater or equal to x: $\lceil 1.1 \rceil = 2$, $\lceil 2.0 \rceil = 2$. |
| $\|x\|$ | Indicates the absolute value of x: $\|-17\|=17$, $\|17\|=17$. |
| $\oplus$ | Indicates exclusive OR (modulo-2 addition). |
| min (x, y) | Indicates the minimum of x and y. |
| max (x, y) | Indicates the maximum of x and y. |
| x mod y | Indicates the remainder after dividing x by y: $x \bmod y = x - (y \times \lfloor x/y \rfloor)$. |

Unless otherwise specified, the format of field values is unsigned binary.

Unless indicated otherwise, this standard presents numbers in decimal form. Binary numbers are distinguished in the text by the use of single quotation marks. Hexadecimal numbers are distinguished by the prefix '0x'.

Unless specified otherwise, each field of a packet shall be transmitted in sequence such that the most significant bit (MSB) is transmitted first and the least significant bit (LSB) is transmitted last. The MSB is the left-most bit in the figures in this document. If there are multiple rows in a table, the top-most row is transmitted first. Within a row in a table, the left-most bit is transmitted first.Notations of the form "repetition factor of N" or "repeated N times" mean that a total of N versions of the item are used.

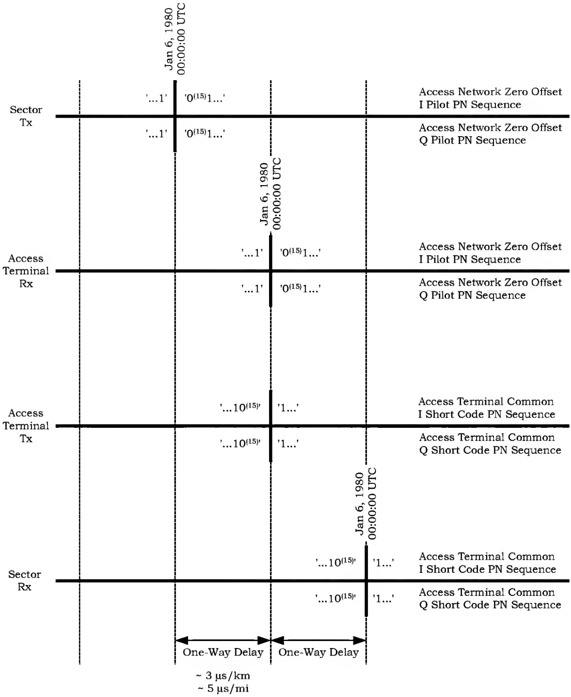
### 1.13 CDMA System Time

All sector air interface transmissions are referenced to a common system-wide timing reference that uses the Global Positioning System (GPS) time, which is traceable to and synchronous with Universal Coordinated Time (UTC). GPS and UTC differ by an integer number of seconds, specifically the number of leap second corrections added to UTC since January 6, 1980. The start of CDMA System Time is January 6, 1980 00:00:00 UTC, which coincides with the start of GPS time.

CDMA System Time keeps track of leap second corrections to UTC but does not use these corrections for physical adjustments to the CDMA System Time clocks.

Figure 1.13-1 shows the relation of CDMA System Time at various points in the system. The access network zero offset pilot PN sequences (as defined in 9.3.1.3.4) and the access terminal common short code PN sequences (as defined in 9.2.1.3.8.1) for the I and Q channels are shown in their initial states at the start of CDMA System Time. The initial state of the access network zero offset pilot PN sequences, both I and Q, is that state in which the next 15 outputs of the pilot PN sequence generator are '0'. The initial state of the access terminal common short code PN sequences, both I and Q, is that state in which the output of the short code PN sequence generator is the '1' following 15 consecutive '0' outputs.

From Figure 1.13-1, note that the CDMA System Time at various points in the transmission and the reception processes is the absolute time referenced at the access network antenna offset by the one-way or round-trip delay of the transmission, as appropriate. Time measurements are referenced to the transmit and receive antennas of the access network and the RF connector of the Access Terminal. The precise zero instant of CDMA System Time is the midpoint between the '1' prior to the 15 consecutive '0' outputs and the immediate succeeding '0' of the access network zero offset pilot PN sequences.



**Notes:**

(1) Time measurements are made at the antennas of Sectors and the RF connectors of the Access Terminals.

(2) $0^{(n)}$ denotes a sequence of n consecutive zeroes.

**Figure 1.13-1. CDMA System Time Line**

## 1.14 Revision Number

Access terminals and access networks complying with the requirements of this specification shall set their revision number to 0x01.

No text.

# 2 DEFAULT SIGNALING APPLICATION

## 2.1 Introduction

### 2.1.1 General Overview

The Default Signaling Application encompasses the Signaling Network Protocol (SNP) and the Signaling Link Protocol (SLP). Protocols in each layer use SNP to exchange messages. SNP is also used by application specific control messages.

SNP provides a single octet header that defines the Type of the protocol with which the message is associated. SNP uses the Type field to route the message to the appropriate protocol.

SLP provides message fragmentation, reliable and best-effort message delivery and duplicate detection for messages that are delivered reliably.

The relationship between SNP and SLP is illustrated in Figure 2.1.1-1.

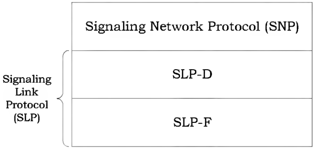

Signaling Network Protocol (SNP)
SLP-D
SLP-F
Signaling Link Protocol (SLP)

**Figure 2.1.1-1. Default Signaling Layer Protocols**

The Signaling Link Protocol consists of two sub-layers, the delivery layer, SLP-D, and the fragmentation layer, SLP-F.

### 2.1.2 Data Encapsulation

Figure 2.1.2-1 and Figure 2.1.2-2 illustrate the relationship between a message, SNP packets, SLP packets, and Stream Layer payloads. Figure 2.1.2-1 shows a case where SLP does not fragment the SNP packet. Figure 2.1.2-2 shows a case where the SLP fragments the SNP packet into more than one SLP-F payload.

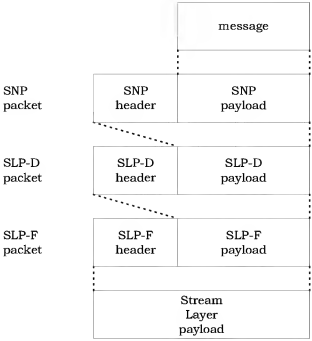


**Figure 2.1.2-1. Message Encapsulation (Non-fragmented)**

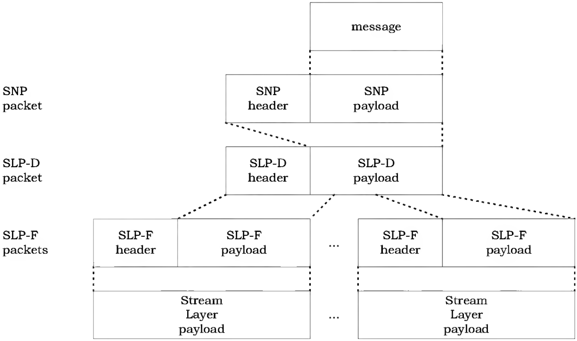


**Figure 2.1.2-2. Message Encapsulation (Fragmented)**

## 2.2 General Signaling Requirements

### 2.2.1 General Requirements

The following requirements are common to all protocols that carry messages using SNP and that provide for message extensibility. The access terminal and the access network shall abide by the following rules when generating and processing any signaling message carried by SNP.

- Messages are always an integer number of octets in length; and, if necessary, include a Reserved field at the end of the message to make them so. The receiver shall ignore the value of the Reserved fields.
- The first field of the message shall be transmitted first. Within each field, the most significant bit of the field shall be transmitted first.
- Message identifiers shall be unambiguous for each protocol Type and for each Subtype for all protocols compatible with the Air Interface, defined by MinimumRevision and above.
- For future revisions, the transmitter shall add new fields only at the end of a message (excluding any trailing Reserved field). The transmitter shall not add fields if their addition makes the parsing of previous fields ambiguous for receivers whose protocol revision is equal to or greater than MinimumRevision.
- The receiver shall discard all unrecognized messages.
- The receiver shall discard all unrecognized fields.
- The receiver shall discard a message if any of the fields in the message is set to a value outside of the defined field range, unless the receiver is specifically directed to ignore this field. A field value is outside of the allowed range if a range was specified with the field and the value is not in this range, or the field is set to a value that is defined as invalid. The receiver shall discard a field in a message if the field is set to a reserved value.

### 2.2.2 Message Information

Each message definition contains information regarding channels on which the message can be transmitted, whether the message requires SLP reliable or best-effort delivery, the addressing modes applicable to the message, and the message priority. This information is provided in the form of a table, an example of which is given in Figure 2.2.2-1.

| | | | |
|---|---|---|---|
| **Channels** | CCsyn | **SLP** | Best Effort |
| **Addressing** | broadcast | **Priority** | 30 |

**Figure 2.2.2-1. Sample Message Information**

The following values are defined:

- Channels: The Physical Layer channel on which this message can be transmitted. Values are:
  - CC for Control Channel (synchronous or asynchronous capsule),
  - CCsyn for Control Channel synchronous capsule,
  - AC for Access Channel,
  - FTC for Forward Traffic Channel, and
  - RTC for Reverse Traffic Channel.
- SLP: Signaling Link Protocol requirements. Values are:
  - Best Effort: the message is sent once and is subject to erasure, and
  - Reliable: erasures are detected and the message is retransmitted one or more times, if necessary.
- Addressing: Addressing modes for the message. Values are:
  - Broadcast if a broadcast address can be used with this message,
  - Multicast if a multicast address can be used with this message, and
  - Unicast if a unicast address can be used with this message.
- Priority: A number between 0 and 255 where lower numbers indicate higher priorities. The priority is used by the Connection Layer (specifically, the Packet Consolidation Protocol) in prioritizing the messages for transmission.

## 2.3 Signaling Network Protocol

### 2.3.1 Overview

The Signaling Network Protocol (SNP) is a message-routing protocol, and routes messages to protocols according to the Type field provided in the SNP header.

The actual protocol indicated by the Type is negotiated during session set-up. For example, Type 0x01 is associated with the Control Channel MAC Protocol. The specific Control Channel MAC Protocol used (and, therefore, the Control Channel MAC protocol generating and processing the messages delivered by SNP) is negotiated when the session is setup.

The remainder of the message following the Type field (SNP header) is processed by the protocol specified by the Type.

### 2.3.2 Primitives and Public Data

#### 2.3.2.1 Commands

This protocol does not define any commands.

#### 2.3.2.2 Return Indications

This protocol does not return any indications.

#### 2.3.2.3 Public Data

The protocol shall make the Type value associated with protocols public.

### 2.3.3 Basic Protocol Numbers

SNP is a protocol associated with the Default Signaling Application. The application subtype for this application is defined in Table 4.2.6.2.1.1-1.

### 2.3.4 Protocol Data Unit

The protocol data unit for this protocol is an SNP packet. Each SNP packet consists of one message sent by a protocol using SNP.

The protocol constructs an SNP packet by adding the SNP header (see 2.3.7) in front of the payload. The structure of the SNP packet is shown in Figure 2.3.4-1.

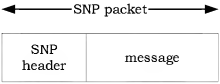


**Figure 2.3.4-1. SNP Packet Structure**

### 2.3.5 Procedures

SNP receives messages for transmission from multiple protocols. SNP shall add the Type field to each message and forward it for transmission to SLP.

SNP receives messages from SLP. SNP shall route these messages to their associated protocols according to the value of the Type field in the SNP header.

If an SNP message is to be transmitted on the Forward Traffic Channel or on the Reverse Traffic Channel, and if a connection is not open, SNP shall issue an *AirLinkManagementProtocol.OpenConnection* command. SNP should queue all messages requiring transmission in the Forward Traffic Channel or in the Reverse Traffic Channel until the protocol receives an *IdleState.ConnectionOpened* indication.

When SNP receives an *SLP.Reset* indication, it shall refrain from passing messages from protocols other than SLP for transmission to SLP until it receives an *SLP.ResetAcked* indication.

### 2.3.6 Type Definitions

Type definitions associated with the default protocol stack are presented in Table 2.3.6-1. The constant name and protocol layer are provided for informational purposes.

**Table 2.3.6-1. Default Protocol Stack Type Values**

| Type | Protocol | Constant Name | Layer |
|---|---|---|---|
| 0x14 | Stream 0 Application | $N_{APP0Type}$ | Application |
| 0x15 | Stream 1 Application | $N_{APP1Type}$ | Application |
| 0x16 | Stream 2 Application | $N_{APP2Type}$ | Application |
| 0x17 | Stream 3 Application | $N_{APP3Type}$ | Application |
| 0x13 | Stream Protocol | $N_{STRType}$ | Stream |
| 0x10 | Session Management Protocol | $N_{SMPType}$ | Session |
| 0x11 | Address Management Protocol | $N_{ADMPType}$ | Session |
| 0x12 | Session Configuration Protocol | $N_{SCPType}$ | Session |
| 0x0a | Air Link Management Protocol | $N_{ALMPType}$ | Connection |
| 0x0b | Initialization State Protocol | $N_{ISPType}$ | Connection |
| 0x0c | Idle State Protocol | $N_{IDPType}$ | Connection |
| 0x0d | Connected State Protocol | $N_{CSPType}$ | Connection |
| 0x0e | Route Update Protocol | $N_{RUPType}$ | Connection |
| 0x0f | Overhead Messages Protocol | $N_{OMPType}$ | Connection |
| 0x09 | Packet Consolidation Protocol | $N_{PCPType}$ | Connection |
| 0x08 | Security Protocol | $N_{SPType}$ | Security |
| 0x05 | Key Exchange Protocol | $N_{KEPType}$ | Security |
| 0x06 | Authentication Protocol | $N_{APType}$ | Security |
| 0x07 | Encryption Protocol | $N_{EPType}$ | Security |
| 0x01 | Control Channel MAC Protocol | $N_{CCMPType}$ | MAC |
| 0x02 | Access Channel MAC Protocol | $N_{ACMPType}$ | MAC |
| 0x03 | Forward Traffic Channel MAC Protocol | $N_{FTCMPType}$ | MAC |
| 0x04 | Reverse Traffic Channel MAC Protocol | $N_{RTCMPType}$ | MAC |
| 0x00 | Physical Layer Protocol | $N_{PHYType}$ | Physical |

### 2.3.7 SNP Header

The SNP shall place the following header in front of every message that it sends:

| Field | Length (bits) |
|---|---|
| Type | 8 |

Type Protocol Type. This field shall be set the Type value for the protocol associated with the encapsulated message.

### 2.3.8 Interface to Other Protocols

#### 2.3.8.1 Commands

This protocol issues the following command:

*AirLinkManagementProtocol.OpenConnection*

#### 2.3.8.2 Indications

This protocol registers to receive the following indications:

- *IdleState.ConnectionOpened*
- *SLP.Reset*
- *SLP.ResetAcked*

## 2.4 Signaling Link Protocol

### 2.4.1 Overview

The Signaling Link Protocol (SLP) has two layers: The delivery layer and the fragmentation layer.

The purpose of the SLP delivery layer (SLP-D) is to provide best effort and reliable delivery for SNP packets. SLP-D provides duplicate detection and retransmission for messages using reliable delivery. SLP-D does not ensure in-order delivery of SNP packets.

The purpose of the SLP fragmentation layer (SLP-F) is to provide fragmentation for SLP-D packets.

### 2.4.2 Primitives and Public Data

#### 2.4.2.1 Commands

This protocol does not define any commands.

#### 2.4.2.2 Return Indications

This protocol returns the following indications:

- *Reset*
- *ResetAcked*

#### 2.4.2.3 Public Data

- None.

### 2.4.3 Basic Protocol Numbers

SLP is a protocol associated with the default signaling application. The application subtype for this application is defined in Table 4.2.6.2.1.1-1.

### 2.4.4 Protocol Data Unit

The protocol data units of this protocol are an SLP-D packet and an SLP-F packet.

### 2.4.5 Procedures

Unless explicitly specified, SLP requirements for the access terminal and the access network are identical; and are, therefore, presented in terms of sender and receiver.

#### 2.4.5.1 Reset

SLP can only be reset at the initiative of the access network. To reset SLP, the access network shall perform the following:

- The access network shall initialize its data structures as described in 2.4.5.3.2 and 2.4.5.2.3.2,

- The access network shall return a *Reset* indication, and
- The access network shall send a Reset message.

Upon receiving a Reset message, the access terminal shall validate the message sequence number as defined in 10.6. If the message is valid, the access terminal shall respond with a ResetAck message and shall initialize its data structures as described in 2.4.5.3.2 and 2.4.5.2.3.2. If the message sequence number of the Reset message is not valid, the access terminal shall discard the message.

The SLP protocol in the access network shall return a *ResetAcked* indication when it receives a ResetAck message with a MessageSequence field equal to the MessageSequence sent in the Reset message. The access network shall increment the sequence number for every Reset message it sends.

The access terminal shall initialize the reset receive pointer used to validate Reset messages (see 10.6) to 0 when the protocol receives a *SessionManagement.BootCompleted* indication.

### 2.4.5.2 Delivery Layer Procedures

#### 2.4.5.2.1 General Procedures

These procedures apply to both the best effort and reliable delivery.

##### 2.4.5.2.1.1 Transmitter Requirements

The transmitter shall take the packet from the upper layer and add the SLP-D header.

The transmitter shall forward the resulting SLP-D packet to the SLP fragmentation layer.

##### 2.4.5.2.1.2 Receiver Requirements

The receiver shall forward the AckSequenceNumber field of the SLP-D header to the co-located transmitter (see 2.4.5.2.3.3.1).

#### 2.4.5.2.2 Best Effort Delivery Procedures

##### 2.4.5.2.2.1 Transmitter Requirements

The transmitter shall set the SequenceValid field of a best-effort SLP-D packet to ‘0’.

##### 2.4.5.2.2.2 Receiver Requirements

The receiver shall forward the SLP-D payload to the upper layer.

#### 2.4.5.2.3 Reliable Delivery Procedures

##### 2.4.5.2.3.1 Overview

SLP-D is an Ack-based protocol with a sequence space of *S*=3 bits.

SLP-D maintains the following variables for reliable delivery SLP-D packet payloads:

- ***V(S)*** The sequence number of the next SLP-D packet to be sent.

- ***V(N)*** The sequence number of the next expected SLP-D packet.
- **Rx** A $2^S$ bit vector. Rx[*i*] = '1' if the SLP-D packet with sequence number *i* was received.

#### 2.4.5.2.3.2 Initialization

When SLP-D is initialized or reset it shall perform the following:

- Set the send state variable *V(S)* to zero in the transmitter.
- Set the receive state variable *V(N)* to zero in the receiver.
- Set Rx[i] to '0' for i = 0…$2^S$-1.
- Clear the retransmission and resequencing buffers.
- Discard any SLP-D packets queued for retransmission.

When SLP-D is initialized or is reset, the sender shall begin sending SLP-D packets with an initial SequenceNumber of 0.

The access terminal and the access network shall perform the initialization procedure if the protocol receives a *ReverseTrafficChannelMAC.LinkAcquired* indication.

#### 2.4.5.2.3.3 Data Transfer

All operations and comparisons performed on SLP-D packet sequence numbers shall be carried out in unsigned modulo $2^S$ arithmetic. For any SLP-D packet sequence number *N*, the sequence numbers in the range [$N+1$, $N+2^{S-1}-1$] shall be considered greater than *N* and the numbers in the range [$N-2^{S-1}$, $N-1$] shall be considered smaller than *N.*

##### 2.4.5.2.3.3.1 Transmit Procedures

The transmitter shall set the SequenceValid field of a reliable-delivery SLP-D packet to '1'.

The transmitter shall acknowledge each reliable-delivery SLP-D packet that its co-located receiver received. The transmitter shall send an acknowledgment within $T_{SLPSDUAck}$ seconds of the receiver receiving a reliable-delivery SLP-D packet. The transmitter acknowledges the received SLP-D packet by setting the AckSequenceNumber field of a transmitted SLP-D packet to the SequenceNumber field of the SLP-D packet being acknowledged, and by setting the AckSequenceValid field to '1'. The transmitter may use the AckSequenceNumber field of an SLP-D it is transmitting; or, if none is available within the required acknowledgment time, it shall transmit an SLP-D header-only SLP-D packet carrying the acknowledgment. The SLP-D header-only SLP-D packet shall be sent as a best-effort SLP-D packet.

Acknowledging an SLP-D packet with sequence number *N* does not imply an acknowledgement for an SLP-D packet with a sequence number smaller than *N.*

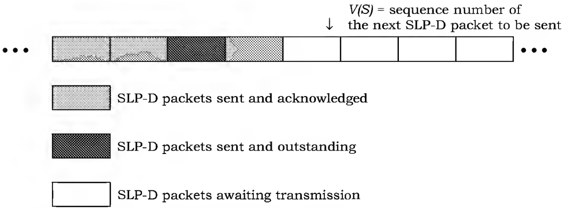

V(S) = sequence number of the next SLP-D packet to be sent
SLP-D packets sent and acknowledged
SLP-D packets sent and outstanding
SLP-D packets awaiting transmission

**Figure 2.4.5.2.3.3.1-1. SLP-D Transmit Sequence Number Variable**

The transmitter shall maintain an *S*-bit variable *V(S)*. The sequence number field (SequenceNumber) in each new SLP-D packet transmitted shall be set to *V(S)*. After transmitting the SLP-D packet, *V(S)* shall be incremented.

If SLP-D has already transmitted $2^{S-1}$ SLP-D packets, SLP-D shall transmit an SDU with sequence number *n*, only after receiving acknowledgments for the SLP-D packets transmitted with sequence number $n$ - $2^{S-1}$ and below, or after determining that these SLP-D packets could not be delivered.

If the transmitter does not receive from its co-located receiver an AckSequenceNumber equal to the SequenceNumber of an outstanding SLP-D packet within $T_{SLPWaitAck}$ seconds, the transmitter shall retransmit the SLP-D packet. The transmitter shall attempt to transmit an SLP-D packet for a maximum of $N_{SLPAttempt}$.

The transmitter shall provide a retransmission buffer for $2^{S-1}$ SLP-D packets. Reliable-delivery SLP-D packets shall be stored in the buffer when they are first transmitted and may be deleted from the buffer, when they are acknowledged or when SLP-D determines that they could not be delivered.

2.4.5.2.3.3.2 Receive Procedures

The SLP-D reliable-delivery receiver shall maintain an *S*-bit variable *V(N)*. *V(N)* contains the sequence number of the next expected SLP-D packet.

The receiver shall maintain a vector Rx with $2^S$ one-bit elements. Rx[*k*] is set to '1' if the SLP-D packet with sequence number *k* has been received.

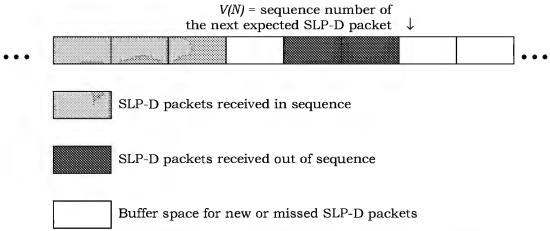

*V(N)* = sequence number of the next expected SLP-D packet
SLP-D packets received in sequence
SLP-D packets received out of sequence
Buffer space for new or missed SLP-D packets

**Figure 2.4.5.2.3.3.2-1. SLP Receive Sequence Number Variables**

For each received SLP-D packet, the receiver shall perform the following actions:

- If a received SLP-D packet has a sequence number k that is smaller than *V(N)* and Rx[*k*] = ‘1’, SLP-D shall discard it as a duplicate.
- If a received SLP-D packet has a sequence number k that is smaller than *V(N)* and Rx[*k*] = ‘0’, SLP-D shall set Rx[k] to ‘1’ and pass the SLP-D payload to the upper layer.
- If a received SLP-D packet has sequence number k that is equal to *V(N)*, SLP-D shall set Rx[*k*] to ‘1’ and Rx[$k+2^{S-1}$] to ‘0’. SLP-D shall set *V(N)* to *k+1* and pass the SLP-D payload to the upper layer.
- If a received SLP-D packet has a sequence number k that is greater than *V(N)*, SLP-D shall set Rx[*k*] to ‘1’, and Rx[*v*] to ‘0’ for all *v* > *k*. SLP-D shall set *V(N)* to *k+1* and pass the SLP-D payload to the upper layer.

2.4.5.3 Fragmentation Layer Procedures

2.4.5.3.1 Overview

SLP-F is a self-synchronizing loss detection protocol with a sequence space of *S* = 6 bits.

SLP-F maintains the following variables for SLP-F packets:

- ***V(S)*** The sequence number of the next SLP-F packet to be sent.
- **Sync** The SLP-F synchronized status flag.

2.4.5.3.2 Initialization

When SLP-F is initialized or reset it shall perform the following:

- Set the send state variable *V(S)* to zero in the transmitter.
- Set Sync to zero.
- Clear the re-assembly buffers.

When SLP-F is initialized or reset, the sender shall begin sending SLP-F packets with an initial SequenceNumber of 0.

The access terminal and the access network shall perform the initialization procedure if the protocol receives a *ReverseTrafficChannelMAC.LinkAcquired* indication.

#### 2.4.5.3.3 Data Transfer

All operations and comparisons performed on SLP-F packet sequence numbers shall be carried out in unsigned modulo $2^S$ arithmetic.

#### 2.4.5.3.4 Sender Requirements

The sender shall construct the SLP-F packet(s) by adding the SLP-F header, defined in 2.4.6.1, in front of each SLP-F payload. The size of each SLP-F packet shall not exceed the current maximum SLP-F packet size.

The sender shall construct the SLP-F payload(s) from an SLP-D packet. If the SLP-D packet exceeds the current maximum SLP-F payload size, then the sender shall fragment the SLP-D packet. If the sender does not fragment the SLP-D packet, then the SLP-D packet is the SLP-F payload. If the sender does fragment the SLP-D packet, then each SLP-D packet fragment is an SLP-F payload.

If the SLP-F payload contains the beginning of an SLP-D packet, then the sender shall set the SLP-F header Begin field to '1'; otherwise, the sender shall set the SLP-F header Begin field to '0'.

If the SLP-F payload contains the end of an SLP-D packet, then the sender shall set the SLP-F header End field to '1'; otherwise, the sender shall set the SLP-F header End field to '0'

The sender shall set the SLP-F SequenceNumber field to *V(S)*.

If the SLP-F payload contains a complete SLP-D packet, then the sender shall not include the SLP-F header Begin, End and SequenceNumber fields; otherwise, the sender shall include the SLP-F header Begin, End and SequenceNumber fields.

The sender shall increment the *V(S)* each time it sends a new SLP-F packet.

#### 2.4.5.3.5 Receiver Requirements

The receiver shall maintain a re-assembly buffer to which it writes the SLP-F payloads when the Sync variable of the SLP-F protocol is equal to 1. The receiver shall perform the following in the order specified:

- If the SLP-F header Fragmented field is '0', then the receiver shall assume the SLP-F header Begin field is '1', the SLP-F header End field is '1' and the SLP-F header SequenceNumber is '0'.
- If the SequenceNumber of the current SLP-F packet is not one greater than SequenceNumber of the last SLP-F packet whose payload was written to the re-assembly buffer, then the receiver shall discard the contents of the re-assembly buffer and shall set the Sync flag to '0'.

- If the Begin field is ‘1’, then the receiver shall discard the contents of the re-assembly buffer and set the Sync flag to ‘1’.
- If the Sync flag is ‘1’, then the receiver shall write the SLP-F payload to the re-assembly buffer, otherwise the receiver shall discard the SLP-F payload.
- If the End field is ‘1’, then the receiver shall pass the contents of the re-assembly buffer to the upper layer and set the Sync flag to ‘0’.

### 2.4.6 Header Formats

The combined SLP-D and SLP-F header length, *x*, is such that

$x$ modulo 8 = 6.

#### 2.4.6.1 SLP-F Header

The SLP-F header length, *x*, is such that

$x$ modulo 8 = 5; if the SLP-F payload contains an SLP-D packet with SLP-D header,

$x$ modulo 8 = 6; if the SLP-F payload contains an SLP-D packet without SLP-D header,

The SLP-F header has the following format:

| Field | Length(bits) |
|---|---|
| Reserved | 4 |
| Fragmented | 1 |
| Begin | 0 or 1 |
| End | 0 or 1 |
| SequenceNumber | 0 or 6 |
| OctetAlignmentPad | 0 or 1 |

Reserved The sender shall set this field to zero. The receiver shall ignore this field.

Fragmented SLP-F header fragmentation indicator. If the rest of the SLP-F header is included, then the sender shall set this field to ‘1’; otherwise, the sender shall set this field to ‘0’. If the SLP-F payload contains a complete SLP-D packet, the sender shall not include the rest of the SLP-F header; otherwise, the sender shall include the rest of the SLP-F header.

Begin Start of SLP-D packet flag. The sender shall only include this field if the Fragmented field is set to ‘1’. If this SLP-F payload contains the beginning of an SLP-D packet, then the sender shall set this field to ‘1’; otherwise, the sender shall set this field to ‘0’.

End End of SLP-D packet flag. The sender shall only include this field if the Fragmented field is set to '1'. If this SLP-F payload contains the end of an SLP-D packet, the sender shall set this field to '1'; otherwise, the sender shall set this field to '0'.

SequenceNumber SLP-F packet sequence number. The sender shall only include this field if the Fragmented field is set to '1'. The sender shall increment this field for each new SLP-F packet sent.

OctetAlignmentPad Octet alignment padding. The sender shall include this field and set it to '0' if the Fragmented field is set to '1' and Begin field is set to '0'. Otherwise, the sender shall omit this field.

### 2.4.6.2 SLP-D Header

The SLP-D header length, $x$, is such that

$x$ modulo 8 = 1.

The SLP-D header has the following format:

| Field | Length(bits) |
|---|---|
| FullHeaderIncluded | 1 |
| AckSequenceValid | 0 or 1 |
| AckSequenceNumber | 0 or 3 |
| SequenceValid | 0 or 1 |
| SequenceNumber | 0 or 3 |

FullHeaderIncluded SLP-D header included flag. If the rest of SLP-D header is included, then the sender shall set this field to '1'; otherwise, the sender shall set this field to '0'. If the sender is either sending or acknowledging a reliable-delivery SLP-D payload, then the sender shall include the rest of the SLP-D header; otherwise, the sender shall not include the rest of the SLP-D header.

AckSequenceValid The sender shall only include this field if the FullHeaderIncluded field is set to '1'. If the AckSequenceNumber field contains a valid value, then the sender shall set this field to '1'; otherwise, the sender shall set this field to '0'. If the sender is acknowledging a reliable-delivery SLP-D payload, then the sender shall include a valid AckSequenceNumber field; otherwise, the sender shall not include a valid AckSequenceNumber field.

AckSequenceNumber

The sender shall only include this field if the FullHeaderIncluded field is set to '1'. If the AckSequenceValid field is set to '1', then the sender shall set this field to the sequence number of the first reliable-delivery

SLP-D payload that has not been acknowledged; otherwise, the sender shall set this field to zero. If the AckSequenceValid field is set to '0', then the receiver shall ignore this field.

SequenceValid The sender shall only include this field if the FullHeaderIncluded field is set to '1'. If the SequenceNumber field contains a valid value, then the sender shall set this field to '1'; otherwise, the sender shall set this field to '0'. If the sender is sending a reliable-delivery SLP-D payload, then the sender shall include a valid SequenceNumber field.

SequenceNumber The sender shall only include this field if the FullHeaderIncluded field is set to '1'. If the SequenceValid field is set to '1', then the sender shall set this field to the sequence number of the reliable SLP-D payload; otherwise, the sender shall set this field to zero. If the SequenceValid field is set to '0', then the receiver shall ignore this field.

### 2.4.7 Message Formats

#### 2.4.7.1 Reset

The Reset message is used by the access network to reset SLP.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| MessageSequence | 8 |

MessageID The access network shall set this field to 0x00.

MessageSequence The access network shall increment this field for every new Reset message it sends.

| | | | | | |
|---|---|---|---|---|---|
| **Channels** | CC | FTC | | **SLP** | Best Effort |
| **Addressing** | | unicast | | **Priority** | 40 |

#### 2.4.7.2 ResetAck

The ResetAck message is used by the access terminal to complete an SLP reset.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| MessageSequence | 8 |

MessageID The access terminal shall set this field to 0x01.

MessageSequence The access terminal shall set this field to the sequence number of the associated Reset message.

| | |
|---|---|
| **Channels** | RTC |
| **Addressing** | unicast |

| | |
|---|---|
| **SLP** | Best Effort |
| Priority | 40 |

### 2.4.8 Protocol Numeric Constants

| Constant | Meaning | Value |
|---|---|---|
| $T_{SLPSDUAck}$ | Time for receiver to acknowledge an arriving reliable-delivery SDU | 200 ms |
| $N_{SLPAttempt}$ | Number of times to retry sending a reliable-delivery SDU | 3 |
| $T_{SLPWaitAck}$ | Retransmission timer for a reliable-delivery SDU | 400 ms |

### 2.4.9 Interface to Other Protocols

#### 2.4.9.1 Commands

This protocol does not issue any commands.

#### 2.4.9.2 Indications

This protocol registers to receive the following indications:

- *ReverseTrafficChannelMAC.LinkAcquired*
- *SessionManagement.BootCompleted*

No text.

# 3 DEFAULT PACKET APPLICATION

## 3.1 Introduction

### 3.1.1 General Overview

The Default Packet Application provides an octet stream that can be used to carry packets between the access terminal and the access network.

The Default Packet Application provides:

- The functionality defined in [1].
- The Radio Link Protocol (RLP), which provides in-order delivery of RLP packets, retransmission, and duplicate detection, thus, reducing the radio link error rate as seen by the higher layer protocols.
- Packet Location Update Protocol, which defines location update procedures and messages in support of mobility management for the Packet Application.
- Flow Control Protocol, which provides flow control for the Default Packet Application Protocol.

The relationship between the Default Packet Application protocols is illustrated in Figure 3.1.1-1.

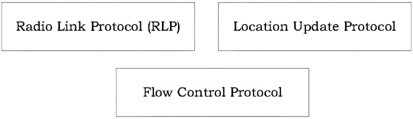


**Figure 3.1.1-1. Default Packet Application Protocols**

### 3.1.2 Data Encapsulation

Figure 3.1.2-1 illustrates the relationship between the octet stream from the upper layer, an RLP packet, and a Stream Layer payload.

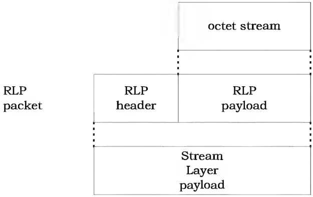


**Figure 3.1.2-1. Default Packet Application Encapsulation**

## 3.2 Radio Link Protocol

### 3.2.1 Overview

The Radio Link Protocol (RLP) provides an octet stream service with an acceptably low erasure rate for efficient operation of higher layer protocols (e.g., TCP). When used as part of the Default Packet Application, the protocol carries an octet stream from the upper layer.

RLP uses Nak-based retransmissions. If the receiver fails to receive octets whose retransmission it requested once, the receiver forwards whatever octets it has to the upper layer and continues reception beyond the missing octets.

### 3.2.2 Primitives and Public Data

#### 3.2.2.1 Commands

This protocol does not define any commands.

#### 3.2.2.2 Return Indications

This protocol does not return any indications.

#### 3.2.2.3 Public Data

- None.

### 3.2.3 Basic Protocol Numbers

RLP is a protocol associated with the default packet application. The application identifier for this application is defined in Table 4.2.6.2.1.1-1.

### 3.2.4 Protocol Data Unit

The transmission unit of this protocol is an RLP packet.

RLP is unaware of higher layer framing; it operates on a featureless octet stream, delivering the octets in the order received from the higher layer.

RLP receives octets for transmission from the higher layer and forms an RLP packet by concatenating the RLP packet header defined in 3.2.6.1 with a number of received contiguous octets. The policy RLP follows in determining the number of octets to send in an RLP packet is beyond the scope of this specification. It is subject to the requirement that an RLP packet shall not exceed the maximum payload length that can be carried by a Stream Layer packet given the target channel and current transmission rate on that channel.

RLP makes use of the Reset, ResetAck, and Nak messages to perform control related operations. When RLP sends these messages it shall use the Signaling Application.

### 3.2.5 Procedures

#### 3.2.5.1 Initialization and Reset

The RLP initialization procedure initializes the RLP variables and data structures in one end of the link. The RLP reset procedure guarantees that RLP state variables on both sides are synchronized. The reset procedure includes initialization.

The access terminal and the access network shall perform the Initialization Procedure defined in 3.2.5.1.1 if the protocol receives an *IdleState.ConnectionOpened* indication.

##### 3.2.5.1.1 Initialization Procedure

When RLP performs the initialization procedure it shall:

- Reset the send state variable *V(S)* to zero,
- reset the receive state variables *V(R)* and *V(N)* to zero,
- clear the resequencing buffer, and
- clear the retransmission queues.

##### 3.2.5.1.2 Reset Procedure

###### 3.2.5.1.2.1 Reset Procedure for the Initiating Side

The side initiating a reset procedure sends a Reset message and enters the RLP Reset State.

Upon entering the RLP Reset state RLP shall:

- Perform the initialization procedure defined in 3.2.5.1.1.
- Ignore all RLP data octets received while in the RLP Reset state.
- If RLP receives a ResetAck message while in the RLP Reset state, it shall send a ResetAck message back and leave the RLP Reset state.

If a ResetAck message is received while RLP is not in the RLP Reset state, the message shall be ignored.

#### 3.2.5.1.2.2 Reset Procedure for the Responding Side

When RLP receives a Reset message, it shall respond with a ResetAck message. After sending the message it shall enter the RLP Reset state, if it was not already in the RLP reset state. Upon entering the RLP Reset state RLP shall:

- Perform the initialization procedure defined in 3.2.5.1.1.
- Ignore all RLP data octets received while in the RLP Reset state.
- When RLP receives a ResetAck message, it shall leave the RLP reset state.

If a ResetAck is received while RLP is not in the RLP Reset state, the message shall be ignored.

### 3.2.5.2 Data Transfer

RLP is a Nak-based protocol with a sequence space of *S* bits, where *S* = 22.

All operations and comparisons performed on RLP packet sequence numbers shall be carried out in unsigned modulo $2^S$ arithmetic. For any RLP octet sequence number *N*, the sequence numbers in the range $[N+1, N+2^{S-1}-1]$ shall be considered greater than *N* and the sequence numbers in the range $[N-2^{S-1}, N-1]$ shall be considered smaller than *N*.

#### 3.2.5.2.1 RLP Transmit Procedures

The RLP transmitter shall maintain an *S*-bit variable *V(S)* for all transmitted RLP data octets (see Figure 3.2.5.2.1-1). *V(S)* is the sequence number of the next RLP data octet to be sent. The sequence number field (SEQ) in each new RLP packet transmitted shall be set to *V(S)*, corresponding to the sequence number of the first octet in the packet. The sequence number of the $i^{th}$ octet in the packet (with the first octet being octet 0) is implicitly given by SEQ+*i*. *V(S)* shall be incremented for each octet contained in the packet.

After transmitting a packet, the RLP transmitter shall start an RLP flush timer for time $T_{RLPFlush}$. If the RLP transmitter sends another packet before the RLP flush timer expires, the RLP transmitter shall reset and restart the timer. If the timer expires, the RLP transmitter shall disable the flush timer and the RLP transmitter shall send an RLP packet containing the octet with sequence number *V(S)*-1. The RLP transmitter should allow sufficient time before deleting a packet transmitted for the first time.

Upon receiving a Nak message, RLP shall insert a copy of the requested octet(s) into its output stream if those octets are available. If the Nak record includes any sequence number greater than or equal to *V(S)*, RLP shall perform the reset procedures specified in 3.2.5.1.2. If the Nak record does not include any sequence number greater than or equal to *V(S)* but the requested octets are not available for retransmissions, RLP shall ignore the Nak.

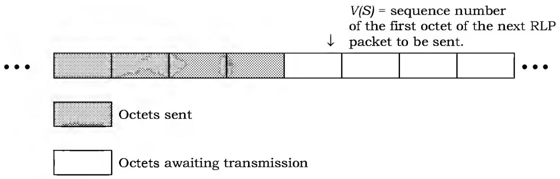

*V(S)* = sequence number
of the first octet of the next RLP
packet to be sent.
Octets sent
Octets awaiting transmission

**Figure 3.2.5.2.1-1. RLP Transmit Sequence Number Variable**

RLP shall assign the following priorities to RLP packets:

- Packet containing re-transmitted octets: 60
- Packet containing octets transmitted for the first time: 70

### 3.2.5.2.2 RLP Receive Procedures

The RLP receiver shall maintain two S-bit variables for receiving, *V(R)* and *V(N)* (see Figure 3.2.5.2.2-1). *V(R)* contains the sequence number of the next octet expected to arrive. *V(N)* contains the sequence number of the first missing octet, as described below.

In addition, the RLP receiver shall keep track of the status of each octet in its resequencing buffer indicating whether the octet was received or not. Use of this status is implied in the following procedures.

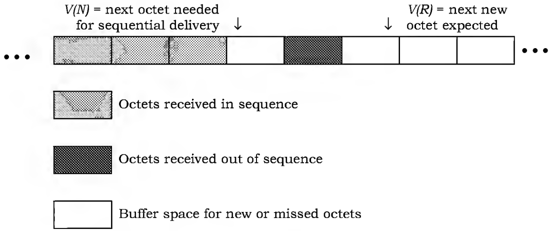

*V(N)* = next octet needed
for sequential delivery
*V(R)* = next new
octet expected
Octets received in sequence
Octets received out of sequence
Buffer space for new or missed octets

**Figure 3.2.5.2.2-1. RLP Receive Sequence Number Variables**

In the following, *X* denotes the sequence number of a received octet. For each received octet, RLP shall perform the following procedures:

- If $X < V(N)$, the octet shall be discarded as a duplicate.
- If $V(N) \le X < V(R)$, and the octet is not already stored in the resequencing buffer, then:

  - RLP shall store the received octet in the resequencing buffer.
  - If $X = V(N)$, RLP shall pass all contiguous octets in the resequencing buffer, from $V(N)$ upward, to the higher layer, and may remove the passed octets from the resequencing buffer. RLP shall then set $V(N)$ to (LAST+1) where LAST is the sequence number of the last octet passed to the higher layer from the resequencing buffer.
- If $V(N) < X < V(R)$, and the octet has already been stored in the resequencing buffer, then the octet shall be discarded as a duplicate.
- If $X = V(R)$, then:
  - If $V(R) = V(N)$, RLP shall increment $V(N)$ and $V(R)$ and shall pass the octet to the higher layer.
  - If $V(R) \neq V(N)$, RLP shall increment $V(R)$ and shall store the octet in the resequencing buffer.
- If $X > V(R)$, then:
  - RLP shall store the octet in the resequencing buffer.
  - RLP shall send a Nak message requesting the retransmission of all missing RLP octets from $V(R)$ to $X$-1, inclusive.
  - RLP shall set $V(R)$ to $X$+1.

RLP shall set a Nak abort timer for each data octet requested in a Nak record for a period of $T_{RLPAbort}$. If a requested octet has not arrived when its Nak abort timer expires, RLP shall pass all octets in the resequencing buffer up to the missing octet, in order of sequence number, to the higher layer. RLP shall skip any missing octets. RLP shall set $V(N)$ to the sequence number of the next missing octet, or to $V(R)$ if there are no remaining missing octets. Further recovery is the responsibility of the upper layer protocols.

### 3.2.6 RLP Packet Header

#### 3.2.6.1 RLP Packet Header

The RLP packet header, which precedes the RLP payload, has the following format:

| Field | Length (bits) |
|---|---|
| SEQ | 22 |

SEQ The RLP sequence number of the first octet in the RLP payload.

### 3.2.7 Message Formats

The messages described in this section control the function of the RLP. These messages are exchanged between the access terminal and the access network using the SNP.

#### 3.2.7.1 Reset

The access terminal and the access network send the Reset message to reset RLP.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |

MessageID The sender shall set this field to 0x00.

| | | | |
|---|---|---|---|
| **Channels** | CC FTC RTC | **SLP** | Reliable |
| **Addressing** | unicast | **Priority** | 50 |

#### 3.2.7.2 ResetAck

The access terminal and the access network send the ResetAck message to complete the RLP reset procedure.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |

MessageID The sender shall set this field to 0x01.

| | | | |
|---|---|---|---|
| **Channels** | CC FTC RTC | **SLP** | Reliable |
| **Addressing** | unicast | **Priority** | 50 |

#### 3.2.7.3 Nak

The access terminal and the access network send the Nak message to request the retransmission of one or more octets.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| NakRequests | 8 |
| NakRequests occurrences of the following three fields: | |
| Reserved | 2 |
| FirstErased | 22 |
| WindowLen | 16 |

MessageID The sender shall set this field to 0x02.

NakRequests The sender shall set this field to the number of Nak requests included in this message. The sender shall include NakRequests occurrences of the following three fields with the message.

Reserved The sender shall set this field to zero. The receiver shall ignore this field.

FirstErased The sender shall set this field to the sequence number of the first RLP octet erased in a sequence of erased octets whose retransmission is requested.

WindowLen The sender shall set this field to the length of the erased window. The receiver shall retransmit all the octets in the range FirstErased to FirstErased+WindowLen-1, inclusive.

| | | | |
|---|---|---|---|
| **Channels** | CC FTC RTC | **SLP** | Best Effort |
| **Addressing** | unicast | **Priority** | 50 |

### 3.2.8 Protocol Numeric Constants

| Constant | Meaning | Value |
|---|---|---|
| $T_{RLPAbort}$ | Time to wait for a retransmission of an octet requested in a Nak message | 500 ms |
| $T_{RLPFlush}$ | Time to wait before retransmitting the last transmitted octet | 300 ms |

### 3.2.9 Interface to Other Protocols

#### 3.2.9.1 Commands

This protocol does not issue any commands.

#### 3.2.9.2 Indications

This protocol registers to receive the following indications:

- *IdleState.ConnectionOpened*

## 3.3 Location Update Protocol

### 3.3.1 Overview

The Location Update Protocol

- Defines location update procedures and messages for mobility management for the Default Packet Application, and
- Negotiates a PDSN selection method and provide data required for PDSN selection.

### 3.3.2 Primitives and Public Data

#### 3.3.2.1 Commands

This protocol does not define any commands.

#### 3.3.2.2 Return Indications

This protocol does not return any indications.

#### 3.3.2.3 Public Data

- None.

### 3.3.3 Basic Protocol Numbers

Packet Location Update Protocol is a protocol associated with the Default Packet Application. The application identifier for this application is defined in Table 4.2.6.2.1.1-1.

### 3.3.4 Protocol Data Unit

The transmission unit of this protocol is a message. This is a control protocol; and, therefore, it does not carry payload on behalf of other layers or protocols.

### 3.3.5 Procedures

#### 3.3.5.1 Access Network Requirements

If the protocol receives an *AddressManagement.SubnetChanged* indication, the access network:

- May send a LocationRequest message to query the Location information.
- May send a LocationAssignment message to update the Location information.

#### 3.3.5.2 Access Terminal Requirements

If the access terminal receives a LocationRequest message, it shall send a LocationResponse message. If the access terminal's current stored LocationValue is not NULL, the access terminal shall set the LocationType, LocationLength, and LocationValue fields in this message to its stored values of these fields. If the access terminal's current stored

LocationValue is equal to NULL, the access terminal shall omit the LocationType, LocationLength, and LocationValue fields in this message.

If the access terminal receives a LocationAssignment message, it shall send a LocationComplete message as follows:

- If the access terminal's current stored Location is not NULL, the access terminal shall set the LocationType, LocationLength, and LocationValue fields of the LocationComplete message to its stored values of these fields. If the access terminal's current stored LocationValue is equal to NULL, the access terminal shall omit the LocationType, LocationLength, and LocationValue fields in this message
- The access terminal shall store the value of the LocationType, LocationLength, and LocationValue fields of the LocationAssignment message in LocationType, LocationLength, and LocationValue variables, respectively.

The access terminal shall set LocationValue to NULL if it receives a *SessionManagement.SessionClosed* indication.

### 3.3.6 Message Formats

### 3.3.6.1 LocationRequest

The access network uses this message to query the access terminal of its Location information.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| TransactionID | 8 |

MessageID The access network shall set this field to 0x03.

TransactionID The access network shall increment this value for each new LocationRequest message sent.

| | | | | |
|---|---|---|---|---|
| **Channels** | CC | FTC | **SLP** | Best Effort |
| **Addressing** | | unicast | **Priority** | 40 |

### 3.3.6.2 LocationResponse

The access terminal sends the LocationResponse message in response to the LocationRequest message.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| TransactionID | 8 |
| LocationType | 8 |
| LocationLength | 0 or 8 |
| LocationValue | 0 or 8 × LocationLength |

MessageID The access terminal shall set this field to 0x04.

TransactionID The access terminal shall set this field the TransactionID field of the corresponding LocationRequest message.

LocationType The access terminal shall set this field to 0 if the value of its stored LocationValue is NULL; otherwise, the access terminal shall set this field to the stored value of LocationType.

LocationLength The access terminal shall not include this field if the value of its stored LocationValue is NULL; otherwise, the access terminal shall set this field to the stored value of LocationLength.

LocationValue The access terminal shall not include this field if the value of its stored LocationValue is NULL; otherwise, the access terminal shall set this field to the stored value of LocationValue.

| | |
|---|---|
| **Channels** | AC RTC |
| **Addressing** | unicast |

| | |
|---|---|
| **SLP** | Reliable[1] Best Effort |
| **Priority** | 40 |

### 3.3.6.3 LocationAssignment

The access network uses this message to update the Location information of the access terminal.

[1] This message is sent reliably when it is sent over the Reverse Traffic Channel.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| TransactionID | 8 |
| LocationType | 8 |
| LocationLength | 8 |
| LocationValue | 8 × LocationLength |

MessageID The access network shall set this field to 0x05.

TransactionID The access network shall increment this value for each new LocationAssignment message sent.

LocationType The access network shall set this field to the type of the location as specified in Table 3.3.6.3-1.

**Table 3.3.6.3-1. LocationType Encoding**

| LocationType | LocationLength | Meaning |
|---|---|---|
| 0x01 | 0x05 | Location compatible with [3] (see Table 3.3.6.3-2) |
| All other values | N/A | Reserved |

LocationLength The access network shall set this field to the length of the LocationValue field in octets as specified in Table 3.3.6.3-1.

LocationValue The access network shall set this field to the Location of type specified by LocationType. If LocationType is set to 0x01, the access network shall set this field as shown in Table 3.3.6.3-2, where SID, NID, and PACKET_ZONE_ID correspond to the current access network.

**Table 3.3.6.3-2. Subfields of LocationValue when LocationType = 0x01**

| Sub-fields of LocationValue | # of bits |
|---|---|
| SID | 15 |
| Reserved | 1 |
| NID | 16 |
| PACKET_ZONE_ID | 8 |

| | | | | |
|---|---|---|---|---|
| **Channels** | CC | FTC | **SLP** | Best Effort |
| **Addressing** | | unicast | **Priority** | 40 |

### 3.3.6.4 LocationComplete

The access terminal sends this message in response to the LocationAssignment message.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| TransactionID | 8 |
| LocationType | 8 |
| LocationLength | 0 or 8 |
| LocationValue | 0 or 8 × LocationLength |

MessageID The access terminal shall set this field to 0x06.

TransactionID The access terminal shall set this field the TransactionID field of the corresponding LocationAssignment message.

LocationType The access terminal shall set this field to 0 if the value of its stored LocationValue is NULL; otherwise, the access terminal shall set this field to the stored value of LocationType.

LocationLength The access terminal shall not include this field if the value of its stored LocationValue is NULL; otherwise, the access terminal shall set this field to the stored value of LocationLength.

LocationValue The access terminal shall not include this field if the value of its stored LocationValue is NULL; otherwise, the access terminal shall set this field to the stored value of LocationValue.

| | | | | | |
|---|---|---|---|---|---|
| **Channels** | AC | RTC | **SLP** | Reliable[2] | Best Effort |
| **Addressing** | | unicast | **Priority** | 40 | |

### 3.3.7 Configuration Attributes

The following complex attribute and default values are defined (see 10.3 for attribute record definition):

[2] This message is sent reliably when it is sent over the Reverse Traffic Channel.

| Field | Length (bits) | Default |
|---|---|---|
| Length | 8 | N/A |
| AttributeID | 8 | N/A |
| One or more of the following record: | | |
| ValueID | 8 | N/A |
| PDSNSelectionType | 8 | 0x00 |
| PDSNSelectionDataLength | 8 | 0x00 |
| PDSNSelectionData | PDSNSelectionDataLength × 8 | N/A |

Length Length of the complex attribute in octets. The access terminal shall set this field to the length of the complex attribute excluding the Length field.

AttributeID The access terminal shall set this field to 0x01.

ValueID The access terminal shall set this field to an identifier assigned to this complex value.

PDSNSelectionType The access terminal shall set this field to the type of the PDSN selection as shown in Table 3.3.7-1.

**Table 3.3.7-1. Encoding of PDSNSelectionType**

| PDSNSelectionType | Meaning |
|---|---|
| 0x00 | The access terminal does not provide the PDSNSelectionData. |
| 0x01 | PDSN selection as specified in [9] |
| All other values | Reserved |

PDSNSelectionDataLength

The access terminal shall set this field to the length of the data provided for PDSN selection as shown in Table 3.3.7-2.

**Table 3.3.7-2. Encoding of PDSNSelectionType, PDSNSelectionDataLength, and PDSNSelectionData**

| PDSNSelectionType | PDSNSelectionDataLength (octets) | PDSNSelectionData |
|---|---|---|
| 0x00 | 0x00 | N/A |
| 0x01 | 0x08 | IMSI |

PDSNSelectionData The access terminal shall set this field to the data needed for PDSN selection with the type specified by PDSNSelectionType as shown in Table 3.3.7-2.

### 3.3.8 Interface to Other Protocols

#### 3.3.8.1 Commands

This protocol does not issue any commands.

#### 3.3.8.2 Indications

This protocol registers to receive the following indications:

- *AddressManagement.Closed*
- *AddressManagement.SubnetChanged*

No text.

## 3.4 Flow Control Protocol

### 3.4.1 Overview

The Flow Control Protocol provides procedures and messages used by the access terminal and the access network to perform flow control for the Default Packet Application Protocol.

This protocol can be in one of the following states:

- Close State: in this state the Default Packet Application does not send or receive any RLP packets.
- Open State: in this state the Default Packet Application can send or receive RLP packets.

Figure 3.4.1-1 and Figure 3.4.1-2 show the state transition diagram at the access terminal and the access network.

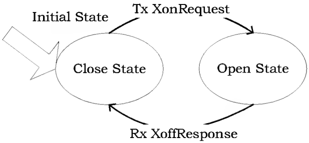


**Figure 3.4.1-1. Flow Control Protocol State Diagram (Access Terminal)**

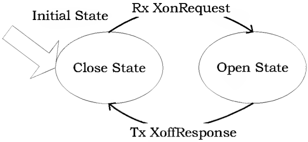


**Figure 3.4.1-2. Flow Control Protocol State Diagram (Access Network)**

### 3.4.2 Primitives and Public Data

#### 3.4.2.1 Commands

This protocol does not define any commands.

#### 3.4.2.2 Return Indications

This protocol does not return any indications.

#### 3.4.2.3 Public Data

- None.

### 3.4.3 Basic Protocol Numbers

Flow Control Protocol is a protocol associated with the Default Packet Application. The application identifier for this application is defined in Table 4.2.6.2.1.1-1.

### 3.4.4 Protocol data Unit

The transmission unit of this protocol is a message. This is a control protocol and, therefore, it does not carry payload on behalf of other layers or protocols.

### 3.4.5 Procedures

#### 3.4.5.1 Transmission and Processing of DataReady Message

The access network may send a DataReady message to indicate that there is data corresponding to this packet application awaiting to be transmitted.

The access terminal shall send a DataReadyAck within the time period specified by $T_{FCResponse}$ of reception of the DataReady message to acknowledge reception of the message.

#### 3.4.5.2 Close State

In this state, the access terminal and the access network shall not send or receive any RLP packets.

##### 3.4.5.2.1 Access Terminal Requirements

The access terminal shall send an XonRequest message when it is ready to exchange RLP packets with the access network. The access terminal should send an XonRequest message when it receives a DataReady from the access network.

The access terminal shall transition to the Open state when it sends an XonRequest message.

##### 3.4.5.2.2 Access Network Requirements

If the access network receives an XonRequest message, it shall

- Send an XonResponse message within the time period specified by $T_{FCResponse}$ of reception of the XonRequest message to acknowledge reception of the message.
- Transition to the Open State.

#### 3.4.5.3 Open State

In this state, the access terminal and the access network may send or receive any RLP packets.

##### 3.4.5.3.1 Access Terminal Requirements

The access terminal may re-send an XonRequest message if it does not receive an XonResponse message an RLP packet within the time period specified by $T_{FCResponse}$ of sending the XonRequest message.

The access terminal may send an XoffRequest message to request the access network to stop sending RLP packets. The access terminal shall transition to the Close state when it receives an XoffResponse message.

The access terminal may re-send an XoffRequest message if it does not receive an XoffResponse message within the time period specified by $T_{FCResponse}$ of sending the XoffRequest message.

##### 3.4.5.3.2 Access Network Requirements

If the access network receives an XoffRequest message, it shall

- Send an XoffResponse message within the time period specified by $T_{FCResponse}$ of reception of XoffRequest message to acknowledge reception of the message.
- Transition to the Close State.

### 3.4.6 Message Formats

#### 3.4.6.1 XonRequest

The access terminal sends this message to request transition to the Open State.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |

MessageID The access terminal shall set this field to 0x07.

| | | | |
|---|---|---|---|
| **Channels** | AC RTC | **SLP** | Best Effort |
| **Addressing** | unicast | **Priority** | 40 |

#### 3.4.6.2 XonResponse

The access network sends this message to acknowledge reception of the XonRequest message.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |

MessageID The access network shall set this field to 0x08.

| | | | |
|---|---|---|---|
| **Channels** | CC FTC | **SLP** | Best Effort |
| **Addressing** | unicast | **Priority** | 40 |

### 3.4.6.3 XoffRequest

The access terminal sends this message to request transition to the Close State.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |

MessageID The access terminal shall set this field to 0x09.

| | | | |
|---|---|---|---|
| **Channels** | AC RTC | **SLP** | Best Effort |
| **Addressing** | unicast | **Priority** | 40 |

### 3.4.6.4 XoffResponse

The access network sends this message to acknowledge reception of the XoffRequest message.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |

MessageID The access network shall set this field to 0x0a.

| | | | |
|---|---|---|---|
| **Channels** | CC FTC | **SLP** | Best Effort |
| **Addressing** | unicast | **Priority** | 40 |

### 3.4.6.5 DataReady

The access network sends this message to indicate that there is data corresponding to this packet application awaiting to be transmitted.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| TransactionID | 8 |

MessageID The access network shall set this field to 0x0b.

TransactionID The access network shall increment this value for each new DataReady message sent.

| | | | | |
|---|---|---|---|---|
| **Channels** | CC FTC | | **SLP** | Best Effort |
| **Addressing** | unicast | | **Priority** | 40 |

### 3.4.6.6 DataReadyAck

The access terminal sends this message to acknowledge reception of a DataReady message.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| TransactionID | 8 |

MessageID The access terminal shall set this field to 0x0c.

TransactionID The access terminal shall set this value to the value of the TransactionID field of the corresponding DataReady message.

| | | | | |
|---|---|---|---|---|
| **Channels** | AC RTC | | **SLP** | Best Effort |
| **Addressing** | unicast | | **Priority** | 40 |

## 3.5 Configuration Messages

The Default Packet Application uses the Generic Configuration Protocol for configuration of the attribute listed in 3.3.7.

### 3.5.1 ConfigurationRequest

The sender sends the ConfigurationRequest message to request the configuration of one or more parameters for the Default Packet Application. The ConfigurationRequest message format is given as part of the Generic Configuration Protocol (see 10.7).

The sender shall set the MessageID field of this message to 0x50.

| | | | |
|---|---|---|---|
| **Channels** | FTC RTC | **SLP** | Reliable |
| **Addressing** | unicast | **Priority** | 40 |

### 3.5.2 ConfigurationResponse

The sender sends the ConfigurationResponse message to select one of the parameter settings offered in an associated ConfigurationRequest message. The ConfigurationResponse message format is given as part of the Generic Configuration Protocol (see 10.7).

The sender shall set the MessageID field of this message to 0x51.

| | | | |
|---|---|---|---|
| **Channels** | FTC RTC | **SLP** | Reliable |
| **Addressing** | unicast | **Priority** | 40 |

No text.

# 4 STREAM LAYER

## 4.1 Introduction

### 4.1.1 General Overview

The Stream Layer provides the following functions:

- Multiplexing of application streams for one access terminal. Stream 0 is always assigned to the Signaling Application. The other streams can be assigned to applications with different QoS (Quality of Service) requirements, or other applications.
- Provision of configuration messages that map applications to streams.

The Stream Layer uses the Stream Layer Protocol to provide these functions.

### 4.1.2 Data Encapsulation

Figure 4.1.2-1 illustrates the relationship between an Application Layer packet, a Stream Layer packet and a Session Layer payload.

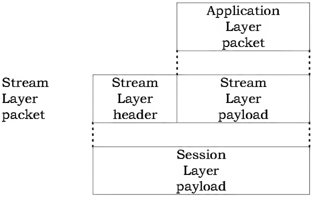


**Figure 4.1.2-1. Stream Layer Encapsulation**

## 4.2 Default Stream Protocol

### 4.2.1 Overview

The Default Stream Protocol provides the Stream Layer functionality. This protocol provides the ability to multiplex up to 4 application streams. Stream 0 is always reserved for a Signaling Application, and, by default, is assigned to the Default Signaling Application. By default, Stream 1 is assigned to the Default Packet Application.

This protocol uses the Generic Configuration Protocol (see 10.7) to define the format and processing of the configuration messages that map applications to streams.

The header added by this protocol is 2 bits in length. If *x* bits is the length of the payload presented to the Stream Layer, *x* shall satisfy

$x$ modulo 8 = 6.

### 4.2.2 Primitives and Public Data

#### 4.2.2.1 Commands

This protocol does not define any commands.

#### 4.2.2.2 Return Indications

This protocol does not return any indications.

#### 4.2.2.3 Public Data

- None.

### 4.2.3 Basic Protocol Numbers

The Type field for this protocol is one octet, set to $N_{STRType}$.

The Subtype field for this protocol is two octets set to $N_{STRDefault}$.

### 4.2.4 Protocol Data Unit

The protocol data unit for this protocol is a Stream Layer Packet.

This protocol receives application packets for transmission from up to four different applications. The protocol adds the Stream header defined in 4.2.6.1 in front of each application packet and forwards it for transmission to the Session Layer.

All Stream Layer packets forwarded to the Session Layer shall be octet aligned.

The protocol receives Stream Layer packets from the Session Layer and removes the Stream Layer header. The application packet obtained in this manner is forwarded to the application indicated by the Stream field of the Stream Layer header.

The structure of the Stream Layer packet is shown in Figure 4.2.4-1

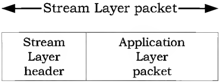


**Figure 4.2.4-1. Stream Layer Packet Structure**

### 4.2.5 Procedures

The access terminal and the access network may use the ConfigurationRequest and ConfigurationResponse messages to select the applications carried by each stream. When the access terminal and the access network use these messages, they shall process them according to the requirements presented in the Generic Configuration Protocol (see 10.7).

Applications can be mapped to the different streams during the AT Initiated State of the Session Configuration Protocol (see 5.4.5.5) as well as during the AN Initiated State of that protocol (see 5.4.5.6).

The ConfigurationRequest and ConfigurationResponse messages may be exchanged only when the session is set-up. The StreamConfiguration attribute and the default values for this attribute are presented in 4.2.6.2.1.1.

### 4.2.6 Header and Message Formats

### 4.2.6.1 Stream Header

The sender adds the following header in front of every Stream Layer payload (application packet):

| Field | Length(bits) |
|---|---|
| Stream | 2 |

Stream The sender shall set this field to the stream number associated with the application sending the application packet following the header.

### 4.2.6.2 Configuration Messages

The Default Stream Protocol uses the Generic Configuration Protocol to associate an application with a particular stream. The following messages are defined:

### 4.2.6.2.1 ConfigurationRequest

The ConfigurationRequest message format is given as part of the Generic Configuration Protocol (see 10.7).

The MessageID field for this message shall be set to 0x50.

| | | | |
|---|---|---|---|
| **Channels** | FTC RTC | **SLP** | Reliable |
| **Addressing** | unicast | **Priority** | 40 |

The following complex attribute and default values are defined (see 10.3 for attribute record definition):

### 4.2.6.2.1.1 StreamConfiguration

| Field | Length (bits) | Default |
|---|---|---|
| Length | 8 | N/A |
| AttributeID | 8 | N/A |
| One or more of the following record: | | |
| ValueID | 8 | N/A |
| Stream0Application | 16 | 0x0000 |
| Stream1Application | 16 | 0xFFFF |
| Stream2Application | 16 | 0xFFFF |
| Stream3Application | 16 | 0xFFFF |

Length Length of the complex attribute in octets. The access network shall set this field to the length of the complex attribute excluding the Length field.

AttributeID The sender shall set this field to 0x00.

ValueID The sender shall set this field to an identifier assigned to this complex value.

Stream0Application The sender shall set this field to the identifier of the application used over Stream 0.

Stream1Application The sender shall set this field to the identifier of the application used over Stream 1.

Stream2Application The sender shall set this field to the identifier of the application used over Stream 2.

Stream3Application The sender shall set this field to the identifier of the application used over Stream 3.

Sender shall set the last four fields to one of the non-reserved values in Table 4.2.6.2.1.1-1.

**Table 4.2.6.2.1.1-1. Application Subtypes**

| Value | Meaning |
|---|---|
| 0x0000 | Default Signaling Application |
| 0x0001 | Default Packet Application bound to the access network. |
| 0x0002 | Default Packet Application bound to the service network. |
| 0xFFFF | Stream not used |
| All other values are reserved. | |

#### 4.2.6.2.2 ConfigurationResponse

The ConfigurationResponse message format is given as part of the Generic Configuration Protocol (see 10.7).

The MessageID field for this message shall be set to 0x51.

If the responder includes an attribute with this message, it shall set the AttributeID field of the message to the AttributeID field of the ConfigurationRequest message associated with this response and the ValueID field to the ValueID field of one of the complex attribute values offered by the ConfigurationRequest message.

| | | | |
|---|---|---|---|
| **Channels** | FTC RTC | **SLP** | Reliable |
| **Addressing** | unicast | **Priority** | 40 |

### 4.2.7 Protocol Numeric Constants

| Constant | Meaning | Value |
|---|---|---|
| $N_{STRType}$ | Type field for this protocol. | Table 2.3.6-1 |
| $N_{STRDefault}$ | Subtype field for this protocol | 0x0000 |

### 4.2.8 Interface to Other Protocols

#### 4.2.8.1 Commands

This protocol does not issue any commands.

#### 4.2.8.2 Indications

This protocol does not register to receive any indications.

No text.

# 5 SESSION LAYER

## 5.1 Introduction

### 5.1.1 General Overview

The Session Layer contains protocols used to negotiate a session between the access terminal and the access network.

A session is a shared state maintained between the access terminal and the access network, including information such as:

- A unicast address (UATI) assigned to the access terminal,
- the set of protocols used by the access terminal and the access network to communicate over the air-link,
- configuration settings for these protocols (e.g., authentication keys, parameters for Connection Layer and MAC Layer protocols, etc.), and
- an estimate of the current access terminal location.

During a single session the access terminal and the access network can open and close a connection multiple times; therefore,  sessions will be closed rarely, and only on occasions such as the access terminal leaving the coverage area or such as prolonged periods in which the access terminal is unavailable.

The Session Layer contains the following protocols:

- <u>Session Management Protocol:</u> This protocol provides the means to control the activation of the other Session Layer protocols. In addition, this protocol ensures the session is still valid and manages closing of the session.
- <u>Address Management Protocol:</u> This protocol specifies procedures for the initial UATI assignment and maintains the access terminal addresses.
- <u>Session Configuration Protocol</u>: This protocol provides the means to negotiate and provision the protocols used during the session, and negotiates the configuration parameters for these protocols. This protocol uses the procedures and attribute-value formats defined by the Generic Configuration Protocol (see 10.7) for protocol negotiation.

The relationship between the Session Layer protocols is illustrated in Figure 5.1.1-1.

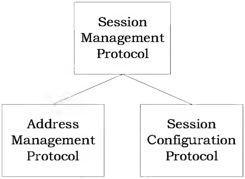

Session Management Protocol
Address Management Protocol
Session Configuration Protocol

**Figure 5.1.1-1. Session Layer Protocols**

### 5.1.2 Data Encapsulation

The Session Layer does not modify transmitted or received packets.

Figure 5.1.2-1 illustrates the relationship between Stream Layer packets, Session Layer packets, and Connection Layer payload.

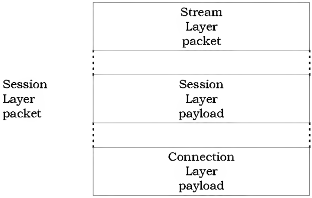


**Figure 5.1.2-1. Session Layer Encapsulation**

## 5.2 Default Session Management Protocol

### 5.2.1 Overview

The Default Session Management protocol provides the means to control the activation of the Address Management Protocol and then the Session Configuration Protocol, in that order, before a session is established. This protocol also periodically ensures that the session is still valid and manages closing the session.

The actual behavior and message exchange in each state of this protocol are mainly governed by protocols that are activated by the Default Session Management Protocol. These protocols return indications, which trigger the state transitions of this protocol.

This protocol can be in one of four states:

- Inactive State: This state applies only to the access terminal. In this state there are no communications between the access terminal and the access network.
- AMP Setup State: In this state the access terminal and access network perform exchanges governed by the Address Management Protocol and the access network assigns a UATI to the access terminal.
- Open State: In this state a session is open.
- Close State: This state applies only to the access network. In this state the access network waits for the close procedure to complete.

Figure 5.2.1-1 provides an overview of the access terminal states and state transitions.

*Failure transitions are not shown*

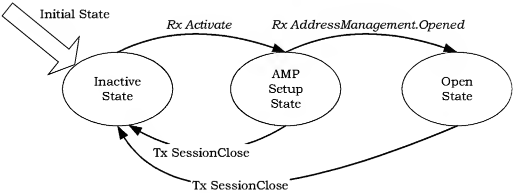


**Figure 5.2.1-1. Session Management Protocol State Diagram (Access Terminal)**

Figure 5.2.1-2 provides an overview of the access network states and state transitions.

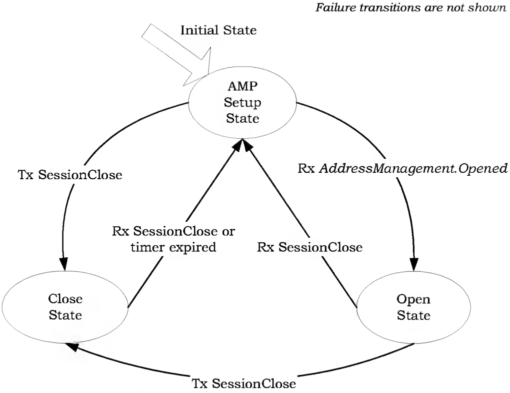


**Figure 5.2.1-2. Session Management Protocol State Diagram (Access Network)**

5.2.2 Primitives and Public Data

5.2.2.1 Commands

This protocol defines the following commands:

- *Activate*
- *Deactivate*

5.2.2.2 Return Indications

This protocol returns the following indications:

- *BootCompleted*
- *SessionOpened*
- *SessionClosed*

#### 5.2.2.3 Public Data

- None.

### 5.2.3 Basic Protocol Numbers

The Type field for the Session Management Protocol is one octet, set to $N_{SMPType}$.

The Subtype field for the Session Management Protocol is two octets, set to $N_{SMPDefault}$.

### 5.2.4 Protocol Data Unit

The transmission unit of this protocol is a message. This is a control protocol and, therefore, it does not carry payload on behalf of other layers or protocols.

This protocol uses the Signaling Application to transmit and receive messages.

### 5.2.5 Procedures

#### 5.2.5.1 Protocol Initialization

This protocol shall be started in the Inactive State for the access terminal.

This protocol shall be started in the Address Management Protocol (AMP) Setup State for the access network.

This protocol does not have any initial configuration requirements.

#### 5.2.5.2 Command Processing

The list of events that causes an *Activate* or *Deactivate* command to be sent to this protocol is outside the scope of this specification.

##### 5.2.5.2.1 Activate

If the access terminal receives the *Activate* command in the Inactive State, it shall transition to the AMP Setup State.

If the access terminal receives the *Activate* command in any state other than the Inactive State, the command shall be ignored.

The access network shall ignore the command.

##### 5.2.5.2.2 Deactivate

If the access terminal receives a *Deactivate* command in the Inactive State, the command shall be ignored.

If the access terminal receives a *Deactivate* command in any state other than the Inactive State, the access terminal shall perform the following:

- Send a SessionClose message to the access network.
- Issue an *AirLinkManagement.CloseConnection* command.
- Issue an *AddressManagement.Deactivate* command.

- Issue a *SessionConfiguration.Deactivate* command.
- Return a *SessionClosed* indication.
- Transition to the Inactive State.

If the access network receives a *Deactivate* command in the Close State, the command shall be ignored.

If the access network receives a *Deactivate* command in any state other than the Close State, the access network shall send a SessionClose message and transition to the Close State.

### 5.2.5.3 Processing the SessionClose Message

If the access terminal receives a SessionClose message in the Inactive State, the message shall be ignored.

If the access terminal receives a SessionClose message in any state other than the Inactive State, the access terminal shall perform the following:

- Send a SessionClose message to the access network.
- Issue an *AirLinkManagement.CloseConnection* command.
- Issue an *AddressManagement.Deactivate* command.
- Issue a *SessionConfiguration.Deactivate* command.
- Return a *SessionClosed* indication.
- Transition to the Inactive State.

If the access network receives a SessionClose message in the Close State, the access network shall process it as specified in 5.2.5.8.

If the access network receives a SessionClose message in any state other than the Close State, the access network shall:

- Issue an *AirLinkManagement.CloseConnection* command.
- Issue an *AddressManagement.Deactivate* command.
- Issue a *SessionConfiguration.Deactivate* command.
- Return a *SessionClosed* indication.
- Transition to the AMP Setup State.

### 5.2.5.4 Processing Failure Indications

The access terminal shall ignore an *AddressManagement.Failed*, or a *SessionConfiguration.Failed* indication, if it receives it in the Inactive State.

If the access terminal receives an *AddressManagement.Failed*, or a *SessionConfiguration.Failed* indication while in any state other than the Inactive State, then the access terminal shall perform the following:

- Send a SessionClose message to the access network.

- Issue an *AirLinkManagement.CloseConnection* command.
- Issue an *AddressManagement.Deactivate* command.
- Issue a *SessionConfiguration.Deactivate* command.
- Return a *SessionClosed* indication.
- The access terminal shall transition to the Inactive State.

If the access network receives an *AddressManagement.Failed*, or a *SessionConfiguration.Failed* indication, the access network shall perform the following:

- Send a SessionClose message to the access terminal.
- Issue an *AirLinkManagement.CloseConnection* command.
- Issue an *AddressManagement.Deactivate* command.
- Issue a *SessionConfiguration.Deactivate* command.
- Return a *SessionClosed* indication.
- Transition to the AMP Setup State.

### 5.2.5.5 Inactive State

This state only applies to the access terminal. In this state there are no communications between the access terminal and the access network. The access terminal does not maintain any session-related state and the access network may be unaware of the access terminal's existence within its coverage area when the access terminal's Session Management Protocol is in this state.

### 5.2.5.6 AMP Setup State

In this state the Session Management Protocol in the access terminal sends an *AddressManagement.Activate* command to the Address Management Protocol and waits for the Address Management Protocol to respond.

#### 5.2.5.6.1 Access Terminal Requirements

Upon entering the AMP Setup State, the access terminal shall send an *AddressManagement.Activate* command to the Address Management Protocol.

If the access terminal receives an *AddressManagement.Opened* indication, it shall perform the following:

- Issue a *SessionConfiguration.Activate* command.
- Return a *BootCompleted* indication.
- Transition to the Open State.

#### 5.2.5.6.2 Access Network Requirements

If the access network receives an *AddressManagement.Opened* indication, it shall perform the following:

- Issue a *SessionConfiguration.Activate* command.
- Return a *BootCompleted* indication.
- Transition to the Open State.

#### 5.2.5.7 Open State

In the Open State the access terminal has an assigned UATI and the access terminal and the access network have configured a session using the Session Configuration Protocol.

If the protocol receives a *SessionConfiguration.SCPChanged* indication, it shall issue a *SessionConfiguration.Activate* command to the selected Session Configuration Protocol.

The access terminal and the access network shall support the keep-alive mechanism defined in 5.2.5.7.1.

##### 5.2.5.7.1 Keep Alive Functions

The access terminal and the access network shall monitor the traffic flowing on the Forward Channel and Reverse Channel, respectively, directed to-or-from the access terminal. If either the access terminal or the access network detects a period of inactivity of at least $T_{SMPClose}/N_{SMPKeepAlive}$ minutes, it may send a KeepAliveRequest message. The recipient of the message shall respond by sending the KeepAliveResponse message. When a KeepAliveResponse message is received, the access terminal shall not send another KeepAliveRequest message for at least $T_{SMPClose}/N_{SMPKeepAlive}$ minutes.

If the access terminal does not detect any traffic from the access network directed to it for a period of at least $T_{SMPClose}$ minutes, it shall perform the following:

- Issue an *AirlinkManagement.CloseConnection* command.
- Issue an *AddressManagement.Deactivate* command.
- Issue a *SessionConfiguration.Deactivate* command.
- Return a *SessionClosed* indication.
- Transition to the Inactive State.

If the access network does not detect any traffic from the access terminal directed to it for a period of at least $T_{SMPClose}$ minutes, it should perform the following:

- Issue an *AirlinkManagement.CloseConnection* command.
- Issue an *AddressManagement.Deactivate* command.
- Issue a *SessionConfiguration.Deactivate* command.
- Return a *SessionClosed* indication.
- Transition to the AMP Setup State.

If the value of $T_{SMPClose}$ is set to zero, the access terminal and the access network shall not send or expect keep-alive messages, and shall disable the transitions occurring as a consequence of not receiving these messages.

### 5.2.5.8 Close State

The Close State is associated only with the protocol in the access network. In this state the protocol in the access network waits for a SessionClose message from the access terminal or an expiration of a timer.

The access network shall set the Close State timer upon entering this state. The value of this timer shall be set to $T_{SMPClose}$ or $T_{SMPMinClose}$, whichever is larger.

When the access network receives a SessionClose message or when the Close State timer expires the protocol shall:

- Issue an *AirLinkManagement.CloseConnection* command.
- Issue an *AddressManagement.Deactivate* command.
- Issue a *SessionConfiguration.Deactivate* command.
- Return a *SessionClosed* indication.
- Transition to the AMP Setup State.

If the access network receives any other Session Management Protocol message from the access terminal using the UATI assigned during this session, it shall discard the message.

## 5.2.6 Message Formats

### 5.2.6.1 SessionClose

The sender sends the SessionClose message to terminate the session.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| CloseReason | 8 |
| MoreInfoLen | 8 |
| MoreInfo | 8 × MoreInfoLen |

MessageID The sender shall set this field to 0x01.

CloseReason The sender shall set this field to the close reason as shown in Table 5.2.6.1-1

**Table 5.2.6.1-1. Encoding of CloseReason Field**

| Field Value | Meaning | MoreInfoLen | MoreInfo |
|---|---|---|---|
| 0x00 | Normal Close | 0 | N/A |
| 0x01 | Close Reply | 0 | N/A |
| 0x02 | Protocol Error | 0 | N/A |
| 0x03 | Protocol Configuration Failure | 3 | Type followed by Subtype |
| 0x04 | Protocol Negotiation Error | variable | zero or more Type followed by Subtype followed by offending attribute records. |
| 0x05 | Session Configuration Failure | 0 | N/A |
| 0x06 | Session Lost | 0 | N/A |
| 0x07 | Session Unreachable | 0 | N/A |
| 0x08 | All session resources busy | 0 | N/A |
| All other values are reserved | | | |

MoreInfoLen Length in octets of the MoreInfo field.

MoreInfo Additional information pertaining to the closure. The format of this field is determined by the particular close reason.

| | | | | |
|---|---|---|---|---|
| **Channels** | CC | AC | FTC | RTC |
| **Addressing** | unicast | | | |

| | |
|---|---|
| **SLP** | Best Effort |
| **Priority** | 40 |

### 5.2.6.2 KeepAliveRequest

The sender sends the KeepAliveRequest to verify that the peer is still alive.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| TransactionID | 8 |

MessageID The sender shall set this field to 0x02.

TransactionID The sender shall increment this value for each new KeepAliveRequest message sent.

| | | | |
|---|---|---|---|
| **Channels** | CC AC FTC RTC | **SLP** | Best Effort |
| **Addressing** | unicast | **Priority** | 40 |

5.2.6.3 KeepAliveResponse

The sender sends the KeepAliveResponse message as an answer to the KeepAliveRequest message.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| TransactionID | 8 |

MessageID The sender shall set this field to 0x03.

TransactionID The sender shall set this value to the value of the TransactionID field of the corresponding KeepAliveRequest message.

| | | | |
|---|---|---|---|
| **Channels** | CC AC FTC RTC | **SLP** | Best Effort |
| **Addressing** | unicast | **Priority** | 40 |

5.2.6.4 Configuration Messages

The Default Session Management Protocol uses the Generic Configuration Protocol for configuration. All configuration messages sent by this protocol shall have their Type field set to $N_{SMPType}$.

The negotiable attributes for this protocol are listed in Table 5.2.6.4-1. The access terminal shall use as defaults the values in Table 5.2.6.4-1 typed in ***bold italics***.

**Table 5.2.6.4-1. Configurable Attributes**

| Attribute ID | Attribute | Values | Meaning |
|---|---|---|---|
| 0xff | $T_{SMPClose}$ | ***0x0CA8*** | Default is 54 hours. |
| | | 0x0000 to 0xFFFF | 0x0000 means disable keep alive messages; all other values are in minutes. |

5.2.6.4.1 ConfigurationRequest

The sender sends the ConfigurationRequest message to request the configuration of one or more parameters for the Session Management Protocol. The ConfigurationRequest message format is given as part of the Generic Configuration Protocol (see 10.7).

The sender shall set the MessageID field of this message to 0x50.

| | | | |
|---|---|---|---|
| **Channels** | FTC RTC | **SLP** | Reliable |
| **Addressing** | unicast | **Priority** | 40 |

#### 5.2.6.4.2 ConfigurationResponse

The sender sends the ConfigurationResponse message to select one of the parameter settings offered in an associated ConfigurationRequest message. The ConfigurationResponse message format is given as part of the Generic Configuration Protocol (see 10.7).

The sender shall set the MessageID field of this message to 0x51.

| | | | |
|---|---|---|---|
| **Channels** | FTC RTC | **SLP** | Reliable |
| **Addressing** | unicast | **Priority** | 40 |

### 5.2.7 Protocol Numeric Constants

| Constant | Meaning | Value |
|---|---|---|
| $N_{SMPType}$ | Type field for this protocol | Table 2.3.6-1 |
| $N_{SMPDefault}$ | Subtype field for this protocol | 0x0000 |
| $N_{SMPKeepAlive}$ | Maximum number of keep alive transactions wthin $T_{SMPClose}$. | 3 |
| $T_{SMPMinClose}$ | Minimum recommended timer setting for Close State | 300 seconds |

### 5.2.8 Interface to Other Protocols

#### 5.2.8.1 Commands Sent

This protocol issues the following commands:

- *AddressManagement.Activate*
- *SessionConfiguration.Activate*
- *AddressManagement.Deactivate*
- *SessionConfiguration.Deactivate*
- *AirLinkManagement.CloseConnection*

#### 5.2.8.2 Indications

This protocol registers to receive the following indications:

- *AddressManagement.Failed*

- *SessionConfiguration.Failed*
- *AddressManagement.Opened*
- *SessionConfiguration.SCPChanged*

## 5.3 Default Address Management Protocol

### 5.3.1 Overview

The Default Address Management Protocol provides the following functions:

- Initial UATI assignment
- Maintaining the access terminal unicast addresse as the access terminal moves between subnets.

This protocol operates in one of three states:

- Inactive State: In this state there are no communications between the access terminal and the access network.
- Setup State: In this state the access terminal and the access network perform a UATIRequest/UATIAssignment/UATIComplete exchange to assign the access terminal a UATI.
- Open State: In this state the access terminal has been assigned a UATI. The access terminal and access network may also perform a UATIRequest/UATIAssignment /UATIComplete or a UATIAssignment/UATIComplete exchange so that the access terminal obtains a new UATI.

The protocol states and the messages and events causing the transition between the states are shown in Figure 5.3.1-1 and Figure 5.3.1-2.

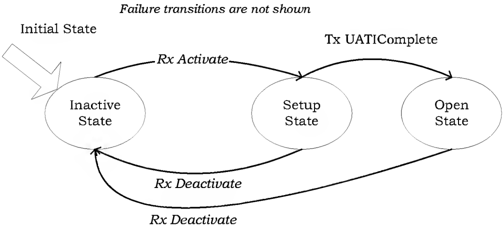


**Figure 5.3.1-1. Address Management Protocol State Diagram (Access Terminal)**

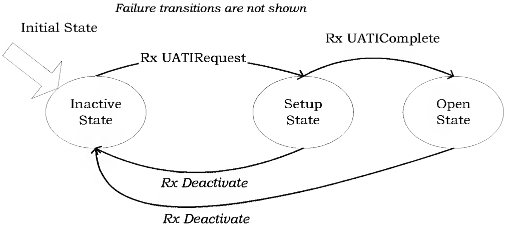


**Figure 5.3.1-2. Address Management Protocol State Diagram (Access Network)**

## 5.3.2 Primitives and Public Data

### 5.3.2.1 Commands

This protocol defines the following command:

- *Activate*
- *Deactivate*
- *UpdateUATI*

### 5.3.2.2 Return Indications

This protocol returns the following indications:

- *Opened*
- *UATIReleased*
- *UATIAssigned*
- *Failed*
- *SubnetChanged*

### 5.3.2.3 Public Data

- ReceiveATIList
- TransmitATI
- SessionSeed

## 5.3.3 Basic Protocol Numbers

The Type field for this protocol is one octet, set to $N_{ADMPType}$.

The Subtype field for this protocol is two octets set to $N_{ADMPDefault}$.

### 5.3.4 Protocol Data Unit

The transmission unit of this protocol is a message. This is a control protocol and, therefore, it does not carry payload on behalf of other layers or protocols.

This protocol uses the Signaling Application to transmit and receive messages.

### 5.3.5 Procedures

#### 5.3.5.1 Protocol Initialization

This protocol shall be started in the Inactive State.

This protocol does not have any initial configuration requirements.

#### 5.3.5.2 Command Processing

##### 5.3.5.2.1 Activate

If the protocol receives the *Activate* command in the Inactive State:

- The access terminal shall transition to the Setup State.
- The access network shall ignore the command.

If the protocol receives the *Activate* command in any state other than the Inactive State, the command shall be ignored.

##### 5.3.5.2.2 Deactivate

If the protocol receives the *Deactivate* command in the Inactive State, the command shall be ignored.

If the protocol receives the *Deactivate* command in any state other than the Inactive State, the protocol shall transition to the Inactive State and return a *UATIReleased* indication.

##### 5.3.5.2.3 UpdateUATI

The access network and access terminal shall ignore the *UpdateUATI* command when it is received in any state other than the Open State.

The access network shall send a UATIAssignment message when it receives an *UpdateUATI* command in the Open State.

The access terminal shall follow the procedures in 5.3.5.6.1.1 to send a UATIRequest message when it receives an *UpdateUATI* command in the Open State.

A comprehensive list of events causing the *UpdateUATI* command is beyond the scope of this specification.

#### 5.3.5.3 UATIAssignment Message Validation

Each time that the access network sends a new UATIAssignment message, it shall increment the value of the MessageSequence field. If the access network is sending the same message multiple times, it shall not change the value of this field between transmissions.

The access terminal shall initialize a receive pointer for the UATIAssignment message validation, *V(R)*, to 255 when it sends a UATIRequest message and ReceiveATIList[$I_{RATI}$].ATI is not set to NULL.

When the access terminal receives a UATIAssignment message, it shall validate the message, using the procedure defined in 10.6 (S is equal to 8). The access terminal shall discard the message if it is stale.

#### 5.3.5.4 Processing HardwareIDRequest message

Upon reception of a HardwareIDRequest message, the access terminal shall respond with a HardwareIDResponse message. The access terminal shall set the HardwareID record of the HardwareIDResponse message to the unique ID that has been assigned to the terminal by the manufacturer.

#### 5.3.5.5 Inactive State

In this state, there are no communications between the access terminal and the access network. The access terminal does not have an assigned UATI, the access network does not maintain a UATI for the access terminal, and may be unaware of the access terminal's existence within its coverage area.

##### 5.3.5.5.1 Access Terminal Requirements

Upon entering the Inactive State, the access terminal shall perform the following:

- Set OldUATI to NULL.
- Set ReceiveATIList[$I_{BATI}$] to <ATIType = '00', ATI = NULL>.
- Set ReceiveATIList[$I_{currentUATI}$] to <ATIType='10', ATI = NULL>.
- Set ReceiveATIList[$I_{newUATI}$] to <ATIType = '10', ATI = NULL>.
- Set ReceiveATIList[$I_{RATI}$] to <ATIType = '11', ATI = NULL>.
- Set TransmitATI to <ATIType = NULL, ATI = NULL>.
- Set UATI to NULL.
- Set UATIColorCode to NULL.
- Set UATISubnetMask to NULL.
- Set SessionSeed to the 32-bit pseudo-random number generated using output of the pseudo random number generator specified in 10.5.

- Disable the DualAddressTimer.

If the access terminal receives an *Activate* command, it shall transition to the Setup State.

#### 5.3.5.5.2 Access Network Requirements

Upon entering the Inactive State, the access network shall perform the following:

- Set the value of the access terminal's UATI to NULL.
- Set the value of the access terminal's UATISubnetMask to NULL.
- Set the value of the access terminal's UATIColorCode to NULL.

The access network shall transition to the Setup State if it receives a UATIRequest message.

### 5.3.5.6 Setup State

In this state, the access terminal sends a request to the access network asking for a UATI and waits for the access network's response.

#### 5.3.5.6.1 Access Terminal Requirements

Upon entering the Setup State the access terminal shall perform the following:

- Set the TransmitATI to
  <ATIType = '11', ATI = SessionSeed>,
- Set ReceiveATIList[$I_{RATI}$] to
  <ATIType = '11', ATI = SessionSeed>.
- Shall follow the procedures in 5.3.5.6.1.1 for sending a UATIRequest message.

A valid (see 5.3.5.3) UATIAssignment message that satisfies either of the following conditions is called a "fresh" UATIAssignment message:

- OverheadParametersUpToDate, provided as the public data of the Overhead Messages Protocol, is equal to 1 and the UATIColorCode field in the message matches the ColorCode, given as public data of the Overhead Messages Protocol, or
- the SubnetIncluded field of the message is equal to '1',

The access terminal shall discard a UATIAssignment message that is not "fresh".

If the access terminal does not receive a "fresh" UATIAssignment message within $T_{ADMPATResponse}$ seconds after receiving an *AccessChannelMAC.TxEnded* indication, it shall return a *Failed* indication and transition to the Inactive State.

If the access terminal receives a "fresh" UATIAssignment message then the access terminal shall perform the following:

- Set the UATIColorCode to the UATIColorCode given in the message.
- Set its UATI and UATISubnetMask as follows:
  - If the message includes the UATI104 field and UATISubnetMask field, the access terminal shall set its UATI to UATI104 | UATI024 and UATISubnetMask to UATISubnetMask field included in the message.

  - Otherwise, the access terminal shall set its UATI to (SectorID[127:24] | UATI024) and UATISubnetMask to SubnetMask where SectorID and SubnetMask are provided as public data of Overhead Messages Protocol.

- Set ReceiveATIList[$I_{RATI}$] to <ATIType = '11', ATI = NULL>.
- Set ReceiveATIList[$I_{currentUATI}$] to <ATIType='10', ATI = (UATIColorCode | UATI[23:0])>.
- Set the TransmitATI to <ATIType='10', ATI = (UATIColorCode | UATI[23:0])>.
- Return an *Opened* indication.
- Return a *UATIAssigned* indication.
- Send a UATIComplete message.
- Transition to the Open State.

5.3.5.6.1.1 Procedures for Sending a UATIRequest message

The access terminal shall follow the following procedures for sending a UATIRequest message:

- If OverheadParametersUpToDate, given as public data by the Overhead Messages Protocol, is equal to 0, the access terminal shall wait until it receives an *OverheadMessages.Updated* indication before it sends a UATIRequest message.
- Otherwise, the access terminal shall send a UATIRequest message without waiting for an *OverheadMessages.Updated* indication.

5.3.5.6.2 Access Network Requirements

When the access network sends a UATIAssignment message, it shall perform the following:

- Access network shall assign a Unicast Access Terminal Identifier (UATI) to the access terminal for the session as follows:
  - Access network may include both UATI104 and UATISubnetMask fields in the UATIAssignment message.
  - Access network may omit the UATI104 and UATISubnetMask fields from the message. In this case, the UATI[127:24] is implicitly assigned to be equal to SectorID[127:24] and UATISubnetMask is implicitly assigned to be SubnetMask, where SectorID and SubnetMask correspond to the sector that has received the UATIRequest message.

When the access network receives the corresponding UATIComplete message with the MessageSequence field of the UATIAssignment message sent, it shall perform the following:

- Return *Opened* indication.
- Return *UATIAssigned* indication.

- Transition to Open State.

If the access network does not receive the corresponding UATIComplete message in response to the UATIAssignment message, it may re-transmit the UATIAssignment message.

### 5.3.5.7 Open State

In this state the access terminal has been assigned a UATI.

#### 5.3.5.7.1 Access Terminal Requirements

If the access terminal receives a *RouteUpdate.IdleHO* indication, and if either of the following two conditions is true, it shall set OldUATI to UATI and follow the procedures in 5.3.5.6.1.1 for sending a UATIRequest message:

- The UATISubnetMask is not equal to the SubnetMask of the sector in the active set, or
- The result of bitwise logical AND of the UATI and its subnet mask specified by UATISubnetMask is different from the result of bitwise logical AND of SectorID and its subnet mask specified by SubnetMask (where SectorID and SubnetMask correspond to the sector in the active set).

Also, if the access terminal receives a *UpdateUATI* command, it shall set OldUATI to UATI and follow the procedures in 5.3.5.6.1.1 for sending a UATIRequest message.

A valid (see 5.3.5.3) UATIAssignment message that satisfies either of the following conditions is called a “fresh” UATIAssignment message:

- OverheadParametersUpToDate, provided as the public data of the Overhead Messages Protocol, is equal to 1 and the UATIColorCode field in the message matches the ColorCode, given as public data of the Overhead Messages Protocol, or
- the SubnetIncluded field of the message equal to ‘1’,

The access terminal shall discard a UATIAssignment message that is not “fresh”.

If the access terminal does not receive a “fresh” UATIAssignment message within $T_{ADMPATResponse}$ seconds after receiving an *AccessChannelMAC.TxEnded* indication, it shall return a *Failed* indication and transition to the Inactive State.

If the access terminal receives a “fresh” UATIAssignment message then the access terminal shall perform the following:

- Set the UATIColorCode to the UATIColorCode given in the message.
- Set its UATI and UATISubnetMask as follows:
  - If the message includes the UATI104 field and UATISubnetMask field, the access terminal shall set its UATI to UATI104 | UATI024 and UATISubnetMask to UATISubnetMask field included in the message.

  - Otherwise, the access terminal shall set its UATI to (SectorID[127:24] | UATI024) and UATISubnetMask to SubnetMask where SectorID and SubnetMask are provided as public data of Overhead Messages Protocol.

- Set ReceiveATIList[$I_{newUATI}$] to <ATIType = '10', ATI = (UATIColorCode | UATI[23:0])>.
- Set the TransmitATI to <ATIType='10', ATI = (UATIColorCode | UATI[23:0])>.
- Return a *UATIAssigned* indication.
- Send a UATIComplete message.
- Reset and start the DualAddress timer with a timeout value of $T_{ADMPDualAddress}$.

The access terminal shall perform the following when the DualAddress timer expires:

- Disable the DualAddress timer.
- Set ReceiveATIList[$I_{currentUATI}$] to ReceiveATIList[$I_{newUATI}$].

If the access terminal receives an *InitializationState.NetworkAcquired* indication and determines that either of the two following conditions is true, it shall return a *Failed* indication and transition to the Inactive State:

- The UATISubnetMask is not equal to the SubnetMask of the sector in the active set, or
- The result of bitwise logical AND of the UATI and its subnet mask specified by UATISubnetMask is different from the result of bitwise logical AND of SectorID and its subnet mask specified by SubnetMask (where SectorID and SubnetMask correspond to the sector in the active set).

#### 5.3.5.7.2 Access Network Requirements

The access network may send a UATIAssignment message at any time in this state. The access network may send a UATIAssignment message if it receives a *RouteUpdate.ActiveSetUpdated* indication, if it receives a *UATIUpdate* command, or in response to a UATIRequest message.

The access network may return a *SubnetChanged* indication and send a UATIAssignment message after reception of a *RouteUpdate.ActiveSetUpdated* indication. The triggers for returning a *SubnetChanged* indication after reception of a *RouteUpdate.ActiveSetUpdated* indication are outside the scope of this specification.

When the access network sends a UATIAssignment message, it shall perform the following:

- Assign a Unicast Access Terminal Identifier (UATI) to the access terminal for the session and include it in a UATIAssignment message.

– If the UATIAssignment message is sent in response to a UATIRequest message, the access network may include both UATI104 and UATISubnetMask. If the access network does not include the UATI104 and UATISubnetMask fields in the message, the UATI[127:24] is implicitly assigned to be equal to SectorID[127:24], where SectorID corresponds to the sector that has received the UATIRequest message.

– Otherwise, the access network shall include both UATI104 and UATISubnetMask fields in the UATIAssignment message.

When the access network receives a UATIComplete message with the MessageSequence field that is equal to the MessageSequence field of the UATIAssignment message that it has sent, it shall return a *UATIAssigned* indication.

If the access network does not receive the UATIComplete message in response to the corresponding UATIAssignment message within a certain time interval that is specified by the access network[3], it should re-transmit the UATIAssignment message.

### 5.3.6 Message Formats

#### 5.3.6.1 UATIRequest

The access terminal sends the UATIRequest message to request that a UATI be assigned or re-assigned to it by the access network.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| TransactionID | 8 |

MessageID The access terminal shall set this field to 0x00.

TransactionID The access terminal shall increment this value modulo 256 for each new UATIRequest message sent.

| | | | |
|---|---|---|---|
| **Channels** | AC | **SLP** | Best Effort |
| **Addressing** | unicast | **Priority** | 10 |

#### 5.3.6.2 UATIAssignment

The access network sends the UATIAssignment message to assign or re-assign a UATI to the access terminal.

[3] The value of this timeout is determined by the access network and specification of the timeout value is outside the scope of this document.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| MessageSequence | 8 |
| Reserved1 | 7 |
| SubnetIncluded | 1 |
| UATISubnetMask | 0 or 8 |
| UATI104 | 0 or 104 |
| UATIColorCode | 8 |
| UATI024 | 24 |
| UpperOldUATILength | 4 |
| Reserved2 | 4 |

MessageID The access network shall set this field to 0x01.

MessageSequence The access network shall set this to 1 higher than the MessageSequence field of the last UATIAssignment message (modulo 256) that it has sent to this access terminal.

Reserved1 The access network shall set this field to zero. The access terminal shall ignore this field.

SubnetIncluded The access network shall set this field to '1' if the UATI104 field and UATISubnetMask fields are included in this message; otherwise, the access network shall set this field to '0'.

UATISubnetMask The access network shall omit this field if SubnetIncluded is set to '0'. If included, the access network shall set this field to the number of consecutive 1's in the subnet mask of the subnet to which the assigned UATI belongs.

UATI104 The access network shall omit this field if SubnetIncluded is set to '0'. If included, the access network shall set this field to UATI[127:24] of the UATI that it is assigning to the access terminal.

UATIColorCode UATI Color Code. The access network shall set this field to the Color Code associated with the subnet to which the UATI belongs.

UATI024 The access network shall set this field to UATI[23:0] of the UATI that it is assigning to the access terminal.

UpperOldUATILength The access network shall set this field the number of least significant bytes of OldUATI[127:24] that the access terminal is to send in the UATIComplete message.

Reserved2 The access network shall set this field to zero. The access terminal shall ignore this field.

| Channels | CC | FTC | | SLP | Best Effort |
|---|---|---|---|---|---|
| Addressing | | unicast | | Priority | 10 |

5.3.6.3 UATIComplete

The access terminal sends this message to notify the access network that it has received the UATIAssignment message.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| MessageSequence | 8 |
| Reserved | 4 |
| UpperOldUATILength | 4 |
| UpperOldUATI | 8 × UpperOldUATILength |

MessageID The access terminal shall set this field to 0x02.

MessageSequence The access terminal shall set this field to the MessageSequence field of the UATIAssignment message whose receipt this message is acknowledging.

Reserved The access terminal shall set this field to zero. The access network shall ignore this field.

UpperOldUATILength The access terminal shall set this field to the length of the UpperOldUATI field in octets.

UpperOldUATI If UpperOldUATILength in the UATIAssignment message whose receipt this message is acknowledging is not zero and OldUATI is not NULL, the access terminal shall set this field to OldUATI[23+UpperOldUATILength×8:24]. Otherwise, the access terminal shall omit this field.

| Channels | AC | RTC | | SLP | Reliable[4] | Best Effort |
|---|---|---|---|---|---|---|

[4] This message is sent reliably when it is sent over the Reverse Traffic Channel.

| | | | |
|---|---|---|---|
| **Addressing** | unicast | **Priority** | 10 |

### 5.3.6.4 HardwareIDRequest

The access network uses this message to query the access terminal of its Hardware ID information.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| TransactionID | 8 |

MessageID The access network shall set this field to 0x03.

TransactionID The access network shall increment this value for each new HardwareRequest message sent.

| | | | |
|---|---|---|---|
| **Channels** | CC FTC | **SLP** | Best Effort |
| **Addressing** | unicast | **Priority** | 40 |

### 5.3.6.5 HardwareIDResponse

The access terminal sends this message in response to the HardwareIDRequest message.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| TransactionID | 8 |
| HardwareIDType | 24 |
| HardwareIDLength | 8 |
| HardwareIDValue | 8×HardwareIDLength |

MessageID The access terminal shall set this field to 0x04.

TransactionID The access terminal shall set this field the TransactionID field of the corresponding HardwareIDRequest message.

HardwareIDType The access terminal shall set this field according to Table 5.3.6.5-1.

**Table 5.3.6.5-1. HardwareIDType encoding**

| HardwareIDType field value | Meaning |
|---|---|
| 0x010000 | Electronic Serial Number (ESN) |
| 0x00NNNN | Hardware ID “NNNN” from [8] |
| 0xFFFFFF | Null |
| All other values | Invalid |

HardwareIDLength If HardwareIDType is not set to 0xFFFFFF, the access terminal shall set this field to the length in octets of the HardwareIDValue field; otherwise the access terminal shall set this field to 0x00.

HardwareIDValue The access terminal shall set this field to the unique ID (specified by HardwareIDType) that has been assigned to the terminal by the manufacturer.

| | | | | | |
|---|---|---|---|---|---|
| **Channels** | AC | RTC | **SLP** | Reliable[5] | Best Effort |
| **Addressing** | | unicast | **Priority** | | 40 |

5.3.7 Protocol Numeric Constants

| Constant | Meaning | Value |
|---|---|---|
| $N_{ADMPType}$ | Type field for this protocol. | Table 2.3.6-1 |
| $N_{ADMPDefault}$ | Subtype field for this protocol | 0x0000 |
| $T_{ADMPATResponse}$ | Time to receive UATIAssignment after sending UATIRequest | 120 seconds |
| $T_{ADMPDualAddress}$ | The duration of time that the access terminal declares an address match if it receives a message that is addressed using either the old or the new UATI | 180 seconds |

5.3.8 Interface to Other Protocols

5.3.8.1 Commands

This protocol does not issue any commands.

[5] This message is sent reliably when it is sent over the Reverse Traffic Channel.

### 5.3.8.2 Indications

This protocol registers to receive the following indications:

- *RouteUpdate.IdleHO*
- *RouteUpdate.ActiveSetUpdated*
- *InitializationState.NetworkAcquired*
- *OverheadMessages.Updated*

## 5.4 Default Session Configuration Protocol

### 5.4.1 Overview

The Default Session Configuration Protocol provides for the negotiation and configuration of the set of protocols used during a session.

This protocol supports two phases of negotiation:

- Access terminal initiated negotiation: In this phase negotiation exchanges are initiated by the access terminal. This phase is used to negotiate the protocols that will be used in the session and negotiate some of the protocols' parameters (e.g., authentication key lengths).
- Access network initiated negotiation: In this phase negotiation exchanges are initiated by the access network. This phase is typically used to override default values used by the negotiated protocols.

This protocol uses the Generic Configuration Protocol procedures and messages when performing the negotiation in each phase (see 10.7). Even if the access terminal requires the use of a Session Configuration Protocol other than the Default Session Configuration Protocol, it shall use the Default Session Configuration Protocol to negotiate the other Session Configuration Protocol.

Example message flow diagrams for an extensive negotiation initiated by the access terminal and a minimal negotiation initiated by the access network are shown in 5.4.9.

Additional protocols may be negotiated without further modifications to the Default Session Configuration Protocol.

This protocol operates in one of four states:

- Inactive State: In this state, the protocol waits for an *Activate* command.
- AT Initiated State: In this state, negotiation is performed at the initiative of the access terminal.
- AN Initiated State: In this state, negotiation is performed at the initiative of the access network.
- Open State: In this state, the access terminal may initiate the session configuration procedure at any time and the access network may request the access terminal to initiate the session configuration at any time.

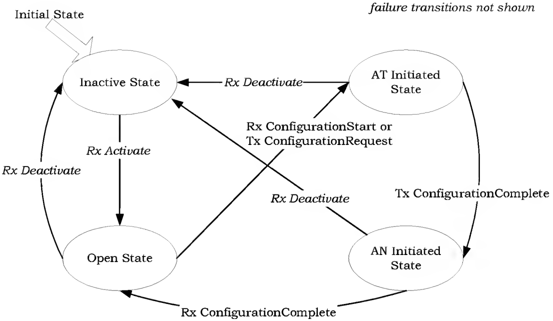


**Figure 5.4.1-1. Session Configuration Protocol State Diagram (Access Terminal)**

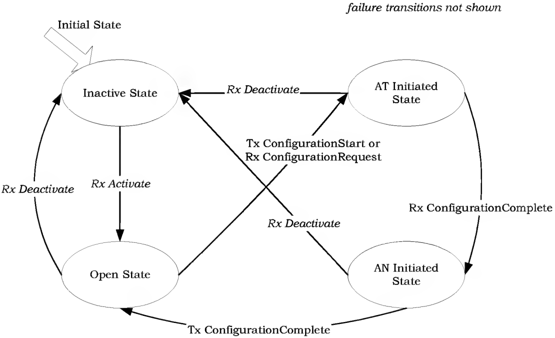


**Figure 5.4.1-2. Session Configuration Protocol State Diagram (Access Network)**

### 5.4.2 Primitives and Public Data

#### 5.4.2.1 Commands

This protocol defines the following commands:

- *Activate*
- *Deactivate*

#### 5.4.2.2 Return Indications

This protocol returns the following indications:

- *SCPChanged*
- *Reconfigured*
- *Failed*

#### 5.4.2.3 Public Data

- Type and subtype of all negotiated protocols
- SessionConfigurationToken

### 5.4.3 Basic Protocol Numbers

The Type field for this protocol is one octet, set to $N_{SCPType}$.

The Subtype field for this protocol is two octets, set to $N_{SCPDefault}$.

### 5.4.4 Protocol Data Unit

The transmission unit of this protocol is a message. This is a control protocol; and, therefore, it does not carry payload on behalf of other layers or protocols.

This protocol uses the Signaling Application to transmit and receive messages.

### 5.4.5 Procedures

#### 5.4.5.1 Protocol Initialization and Configuration

This protocol shall be started in the Inactive State.

This protocol does not have any initial configuration requirements.

#### 5.4.5.2 Processing the Activate Command

If the protocol receives the *Activate* command in the Inactive State, it shall transition to the Open State.

If this command is received in any other state it shall be ignored.

#### 5.4.5.3 Processing the Deactivate Command

If the protocol receives the *Deactivate* command in the Inactive State it shall be ignored.

If the protocol receives this command in the AT Initiated State, AN Initiated State, or Open State, it shall transition to the Inactive State.

#### 5.4.5.4 Inactive State

Upon entering this state, the protocol shall perform the following:

- Set the SessionConfigurationToken to 0x0000.
- Set the protocols and protocol configurations to their default values.

In this state the protocol waits for the *Activate* command. See 5.4.5.2 for processing of the *Activate* command in this state.

#### 5.4.5.5 AT Initiated State

During the AT Initiated State of the Default Session Configuration Protocol the access terminal and the access network use the Generic Configuration Protocol (see 10.7) with the access terminal being the initiator of each exchange. The access terminal and the access network use the ConfigurationRequest/ConfigurationResponse exchange defined in 10.7 to select the protocols that will be used for the session.

Also, the access terminal may request restoring a previously established session in this state.

The default values for all the attributes and protocols shall be the values that were agreed upon prior to entering this state.

The protocol in the access terminal or the access network shall return a *Failed* indication and transition to the Inactive state, if any of the negotiated protocols declares a failure.

##### 5.4.5.5.1 Access Terminal Requirements

If the access terminal chooses to request restoring a prior session, it shall perform the following in the order specified:

- The access terminal shall construct a 32-bit pseudo random number, Nonce.
- The access terminal shall temporarily configure the protocols within the Security Layer with the parameters (i.e., the session key and all the negotiated protocols and attributes in the security layer) associated with the prior session.
- The access terminal shall supply the Nonce, to the security layer of the prior session as if the Nonce is the payload to be transmitted on the Access Channel. The access terminal shall set all the unspecified parameters needed by the protocols in the Security Layer to zero for the purpose of generating this Security Layer Packet.
- The access terminal shall restore the Security Layer to its previous configuration.
- The access terminal shall set the SecurityPacket variable to the Security Layer Packet constructed in the previous step.

- The access terminal shall send the UATI corresponding to the prior session and the SecurityPacket variables as a complex attribute (see 5.4.6.3.2) in a ConfigurationRequest message.

The access terminal may send the access network ConfigurationRequest messages, requesting the use of specific protocols per the Generic Configuration Protocol.

The access terminal shall process the ConfigurationResponse messages it receives per the Generic Configuration Protocol.

Following the receipt of a ConfigurationResponse message, the access terminal may:

- Send another ConfigurationRequest message attempting to negotiate a different protocol for the protocol Type specified in the ConfigurationResponse message.
- Use the protocol configuration procedures defined by the protocol to perform access terminal-initiated parameter configuration.

If after performing access terminal-initiated parameter configuration, the access terminal requires the use of a different protocol for this protocol Type, the access terminal may send the access network a new ConfigurationRequest message.

If the access terminal sends a ConfigurationRequest message specifying a protocol Type for which protocol negotiation procedures were previously executed in this state, the access terminal shall discard all parameters negotiated during that procedure.

If the protocol in access terminal requires no further negotiation of protocols or configuration of negotiated protocols, it shall send a ConfigurationComplete message to the access network and transition to the AN Initiated State.

#### 5.4.5.5.2 Access Network Requirements

If the access network receives a ConfigurationRequest message from the access terminal, it shall process it and shall respond with a ConfigurationResponse message per the Generic Configuration Protocol.

Once the access network sends a ConfigurationResponse message for a particular protocol, it shall be ready to execute the access terminal-initiated configuration procedures that are particular to that protocol.

If the access network receives a ConfigurationRequest message, specifying a protocol Type for which it has previously executed a parameter negotiation procedure, the access network shall discard all parameters negotiated during that procedure.

If the protocol in the access network receives a ConfigurationComplete message, it shall transition to the AN Initiated State.

### 5.4.5.6 AN Initiated State

During the AN Initiated State of the protocol, the access network and the access terminal execute the access network-initiated configuration procedures specified by each negotiated protocol. These procedures typically allow the access network to override default values otherwise used by the access terminal.

If the access network initiates negotiation of an attribute, the default value for the attribute shall be the value agreed upon prior to entering this state.

#### 5.4.5.6.1 Access Terminal Requirements

In this protocol state the access terminal shall be ready to execute the access network-initiated configuration procedures particular to each protocol used during the session.

If the access terminal receives a ConfigurationRequest message from the access network, it shall process it and shall respond with a ConfigurationResponse message according to the Generic Configuration Protocol.

If the access terminal receives a ConfigurationComplete message it shall:

- Issue an *AirlinkManagement.CloseConnection* command.
- Return a *Reconfigured* indication.
- Transition to the Open State.

If as a result of ConfigurationRequest/ConfigurationResponse exchange a non-default Session Configuration Protocol is selected, the access terminal shall return an *SCPChanged* indication.

If as a result of ConfigurationRequest/ConfigurationResponse exchange a PriorSession attribute (with a non-zero Restore field) is agreed upon, the protocols and attributes corresponding to the session specified by the PriorSession attribute shall take effect after the protocol receives a *ConnectedState.ConnectionClosed* indication. Otherwise, the newly negotiated protocols and attributes shall take effect after the protocol receives a *ConnectedState.ConnectionClosed* indication.

#### 5.4.5.6.2 Access Network Requirements

In this protocol state, the access network may execute the access network-initiated configuration procedures that are particular to each protocol used during the session.

If the access network chooses to negotiate a different Session Configuration Protocol, it shall initiate the Session Configuration Protocol selection (i.e., sending a ConfigurationRequest message specifying protocol Type of $N_{SCPType}$) prior to selection of any other protocol.

The access network may set the SessionConfigurationToken field of the ConfigurationComplete message to reflect the selected protocols and the negotiation parameters associated with the negotiated protocols. The rules for setting this field are outside the scope of this specification.

If the protocol in access network requires no further negotiation of protocols or configuration of negotiated protocols, it shall:

- Send a ConfigurationComplete message to the access terminal.
- Issue an *AirlinkManagement.CloseConnection* command.
- Return a *Reconfigured* indication.

- Transition to the Open State.

If as a result of ConfigurationRequest/ConfigurationResponse exchange a non-default Session Configuration Protocol is selected, the access network shall return an *SCPChanged* indication.

If as a result of ConfigurationRequest/ConfigurationResponse exchange a PriorSession attribute (with a non-zero Restore field) is agreed upon, the protocols and attributes corresponding to the session specified by the PriorSession attribute shall take effect after the protocol receives a *ConnectedState.ConnectionClosed* indication. Otherwise, the newly negotiated protocols and attributes shall take effect after the protocol receives a *ConnectedState.ConnectionClosed* indication.

### 5.4.5.7 Open State

#### 5.4.5.7.1 General Requirements

In this protocol state the access terminal and the access network use the negotiated protocols to exchange data and signaling in accordance with the requirements of each protocol.

The protocol in the access network may send a ConfigurationStart message at any time during the Open State to start the negotiation process (e.g., the access network may send this message to negotiate a new stream).

The protocol in the access terminal may send a ConfigurationRequest message at any time during the Open State to start the negotiation process (e.g., the access terminal may send this message to negotiate a new stream).

The protocol in the access terminal transitions to the AT Initiated State when it receives a ConfigurationStart message or when it sends a ConfigurationRequest message.

The protocol in the access network transitions to the AT Initiated State when it sends a ConfigurationStart message or when it receives a ConfigurationRequest message.

### 5.4.6 Message Formats

#### 5.4.6.1 ConfigurationComplete

The sender sends the ConfigurationComplete message to indicate that it has completed the negotiation procedures performed at its initiative.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| TransactionID | 8 |
| SessionConfigurationToken | 0 or 16 |

MessageID The sender shall set this field to 0x00.

TransactionID The access terminal shall increment this value for each new ConfigurationComplete message sent. The access network shall set this value to the value of TransactionID included in the last ConfigurationComplete message received from the access terminal.

SessionConfigurationToken

Session Configuration Token. The access terminal shall omit this field. The access network shall include this field. The access network may set this field to a 16-bit value that reflects the selected protocols and the negotiation parameters associated with the negotiated protocols.

| | | | |
|---|---|---|---|
| **Channels** | FTC RTC | **SLP** | Reliable |
| **Addressing** | unicast | **Priority** | 40 |

### 5.4.6.2 ConfigurationStart

The access network sends this message to start a session configuration process.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |

MessageID The sender shall set this field to 0x01.

| | | | |
|---|---|---|---|
| **Channels** | CC FTC | **SLP** | Best Effort |
| **Addressing** | unicast | **Priority** | 40 |

### 5.4.6.3 Configuration Messages

The Default Session Configuration Protocol uses the Generic Configuration Protocol for configuration. All configuration messages sent by this protocol shall have their Type field set to $N_{SCPType}$.

The following attribute-value pairs are defined (see 10.3 for attribute record format). All attribute fields for the Default Session Configuration Protocol are two octets in length. .

#### 5.4.6.3.1 Protocol Type Attributes

The Protocol Type configurable attributes are listed in Table 5.4.6.3.1-1. All these attributes are simple. The Attribute ID field for all these attributes are two octets in length and the value fields for these attributes are two octets in length

**Table 5.4.6.3.1-1. Protocol Type Configurable Attributes**

| Attribute ID | Attribute | Values | Meaning |
|---|---|---|---|
| 0x00NN | Protocol Type, where NN is the hexadecimal Protocol Type value. | ***0x0000*** | Default Protocol Subtype. |
| | | 0x0000 – 0xFFFF | Protocol Subtype. |

5.4.6.3.2 PriorSession Attribute

The following complex attribute and default values are defined (see 10.3 for attribute record definition):

| Field | Length (bits) | Default |
|---|---|---|
| Length | 8 | N/A |
| AttributeID | 16 | N/A |
| One or more of the following record: | | |
| ValueID | 8 | N/A |
| Restore | 1 | '0' |
| Reserved | 7 | '0000000' |
| UATI | 0 or 128 | N/A |
| SecurityPacketLength | 0 or 8 | N/A |
| SecurityPacket | 0 or SecurityPacketLength × 8 | N/A |

Length Length of the complex attribute in octets. The access terminal shall set this field to the length of the complex attribute excluding the Length field.

AttributeID The access terminal shall set this field to 0x1000.

ValueID The access terminal shall set this field to an identifier assigned to this complex value.

Restore The access terminal shall set this field to '1' if it is requesting to restore a prior session. The access terminal shall set this field to '0' if it is requesting to proceed with the current session configuration and not restore any prior sessions.

Reserved The access terminal shall set this field zero. The access network shall ignore this field.

UATI The access terminal shall include this field only if the Restore field is set to '1'. If included, the access terminal shall set this field to the UATI associated with the prior session.

SecurityPacketLength
The access terminal shall include this field only if the Restore field is set to '1'. If included, the access terminal shall set this field to the length of the SecurityPacket filed in octets.

SecurityPacket The access terminal shall include this field only if the Restore field is set to '1'. If included, the access terminal shall set this field to the SecurityPacket variable which is constructed as specified in 5.4.5.5.1.

#### 5.4.6.3.3 ConfigurationRequest

The sender sends the ConfigurationRequest message to request the configuration of one or more parameters for the Session Configuration Protocol.[6] The ConfigurationRequest message format is given as part of the Generic Configuration Protocol (see 10.7).

The sender shall set the MessageID field of this message to 0x50.

| **Channels** | FTC RTC | **SLP** | Reliable |
|---|---|---|---|
| **Addressing** | unicast | **Priority** | 40 |

#### 5.4.6.3.4 ConfigurationResponse

The sender sends the ConfigurationResponse message to select one of the parameter settings offered in an associated ConfigurationRequest message. The ConfigurationResponse message format is given as part of the Generic Configuration Protocol (see 10.7).

The sender shall set the MessageID field of this message to 0x51.

| **Channels** | FTC RTC | **SLP** | Reliable |
|---|---|---|---|
| **Addressing** | unicast | **Priority** | 40 |

### 5.4.7 Protocol Numeric Constants

[6] Most of the Session Configuration Protocol parameters being configured are the specific (i.e., Subtype) protocols used for each protocol Type.

| Constant | Meaning | Value |
|---|---|---|
| $N_{SCPType}$ | Type field for this protocol | Table 2.3.6-1 |
| $N_{SCPDefault}$ | Subtype field for this protocol | 0x0000 |

### 5.4.8 Interface to Other Protocols

#### 5.4.8.1 Commands

This protocol issues the following command:

- *AirLinkManagement.CloseConnection*

#### 5.4.8.2 Indications

This protocol registers to receive the following indication:

- *ConnectedState.ConnectionClosed*

## 5.4.9 Message Flows

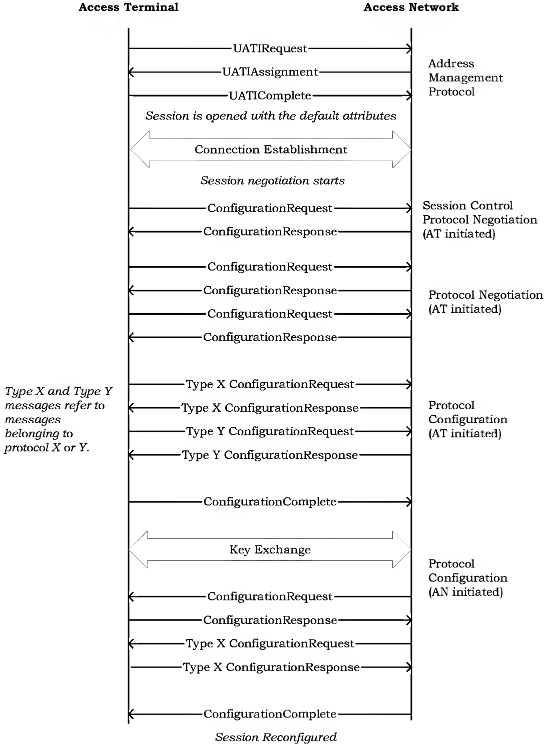

Access Terminal
Access Network
UATIRequest
UATIAssignment
UATIComplete
Address Management Protocol
Session is opened with the default attributes
Connection Establishment
Session negotiation starts
ConfigurationRequest
ConfigurationResponse
Session Control Protocol Negotiation (AT initiated)
ConfigurationRequest
ConfigurationResponse
ConfigurationRequest
ConfigurationResponse
Protocol Negotiation (AT initiated)
Type X ConfigurationRequest
Type X ConfigurationResponse
Type Y ConfigurationRequest
Type Y ConfigurationResponse
Type X and Type Y messages refer to messages belonging to protocol X or Y.
Protocol Configuration (AT initiated)
ConfigurationComplete
Key Exchange
ConfigurationRequest
ConfigurationResponse
Type X ConfigurationRequest
Type X ConfigurationResponse
Protocol Configuration (AN initiated)
ConfigurationComplete
Session Reconfigured

**Figure 5.4.9-1. Default Session Configuration Protocol: Extensive Negotiation Procedure**

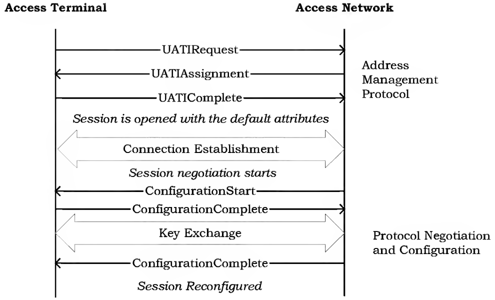


**Figure 5.4.9-2. Default Session Configuration Protocol: Minimal Negotiation Procedure with Key Exchange**

No text.

# 6 CONNECTION LAYER

## 6.1 Introduction

### 6.1.1 General Overview

The Connection Layer controls the state of the air-link, and it prioritizes the traffic that is sent over it.

This section presents the default protocols for the Connection Layer. With the exception of the Overhead Messages Protocol, each of these protocols can be independently negotiated at the beginning of the session.

The access terminal and the access network maintain a connection whose state dictates the form in which communications between these entities can take place. The connection can be either closed or open:

- Closed Connection: When a connection is closed, the access terminal is not assigned any dedicated air-link resources. Communications between the access terminal and the access network are conducted over the Access Channel and the Control Channel.
- Open Connection: When a connection is open, the access terminal can be assigned the Forward Traffic Channel, and is assigned a Reverse Power Control Channel and a Reverse Traffic Channel. Communications between the access terminal and the access network are conducted over these assigned channels, as well as over the Control Channel.

The Connection Layer provides the following connection-related functions:

- Manages initial acquisition of the network.
- Manages opening and closing of connections.
- Manages communications when connection is closed and when a connection is open.
- Maintains approximate access terminal's location in either connection states.
- Manages radio link between the access terminal and the access network when a connection is open.
- Performs supervision both when the connection is open and when it is closed.
- Prioritizes and encapsulates transmitted data received from the Session Layer and forwards it to the Security Layer.
- De-capsulates data received from the Security Layer and forwards it to the Session Layer.

The Connection Layer performs these functions through the following protocols:

- Air Link Management Protocol: This protocol maintains the overall connection state in the access terminal and the access network. The protocol can be in one of three states, corresponding to whether the access terminal has yet to acquire the network (Initialization State), has acquired the network but the connection is closed (Idle State), or has an open connection with the access network (Connected State). This protocol activates one of the following three protocols as a function of its current state.

- Initialization State Protocol: This protocol performs the actions associated with acquiring an access network.

- Idle State Protocol: This protocol performs the actions associated with an access terminal that has acquired the network, but does not have an open connection. Mainly, these are keeping track of the access terminal's approximate location in support of efficient Paging (using the Route Update Protocol), the procedures leading to the opening of a connection, and support of access terminal power conservation.

- Connected State Protocol: This protocol performs the actions associated with an access terminal that has an open connection. Mainly, these are managing the radio link between the access terminal and the access network (handoffs, handled via the Route Update Protocol), and the procedures leading to the close of the connection.

In addition to the above protocols, which deal with the state of the connection, the Connection Layer also contains the following protocols:

- Route Update Protocol: This protocol performs the actions associated with keeping track of an access terminal's location and maintaining the radio link between the access terminal and the access network. This protocol performs supervision on the pilots.

- Overhead Messages Protocol: This protocol broadcasts essential parameters over the Control Channel. These parameters are shared by protocols in the Connection Layer as well as protocols in other layers. This protocol also performs supervision on the messages necessary to keep the Connection Layer functioning.

- Packet Consolidation Protocol: This protocol consolidates and prioritizes packets for transmission as a function of their assigned priority and the target transmission channel.

Figure 6.1.1-1 illustrates the relationship between all the Connection Layer protocols. An arrow between two protocols implies that the source sends commands to the target.

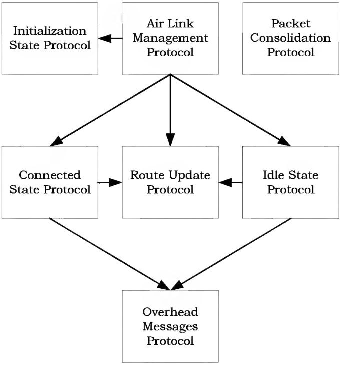

Initialization State Protocol
Air Link Management Protocol
Packet Consolidation Protocol
Connected State Protocol
Route Update Protocol
Idle State Protocol
Overhead Messages Protocol

**Figure 6.1.1-1. Connection Layer Protocols**

The Air Link Management Protocol, its descendants and the Overhead Messages Protocol are control protocols. The Packet Consolidation Protocol operates on transmitted and received data.

### 6.1.2 Data Encapsulation

In the transmit direction, the Connection Layer receives Session Layer packets, adds Connection Layer header(s) and padding, where applicable, and forwards the resulting packet for transmission to the Security Layer.

In the receive direction, the Connection Layer receives Security Layer packets from the Security Layer, and forwards them to the Session Layer after taking off the Connection Layer headers and padding.

Figure 6.1.2-1 and Figure 6.1.2-2 illustrate the relationship between Session Layer packets, Connection Layer packets and Security Layer payloads for Format A (maximum size) and Format B Connection Layer packets.

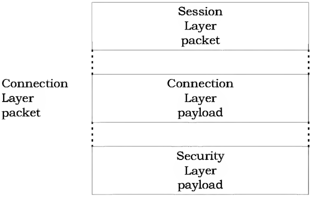


**Figure 6.1.2-1. Connection Layer Encapsulation (Format A)**

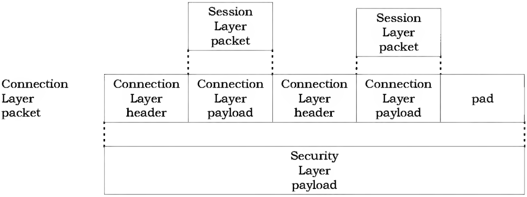


**Figure 6.1.2-2. Connection Layer Encapsulation (Format B)**

## 6.2 Default Air-Link Management Protocol

### 6.2.1 Overview

The Default Air-Link Management Protocol provides the following functions:

- General state machine and state-transition rules to be followed by an access terminal and an access network for the Connection Layer
- Activation and deactivation of Connection Layer protocols applicable to each protocol state
- Mechanism through which access network can redirect access terminal to another network

The actual behavior and message exchange in each state is mainly governed by protocols that are activated by the Default Air-Link Management Protocol. These protocols return indications which trigger the state transitions of this protocol. These protocols also share data with each other in a controlled fashion, by making that data public.

This protocol can be in one of three states:

- Initialization State: In this state the access terminal acquires an access network. The protocol activates the Initialization State Protocol to execute the procedures relevant to this state. The access network maintains a single instance of this state and consequently, executes a single instance of the Initialization State Protocol.
- Idle State: In this state the connection is closed. The protocol activates the Idle State Protocol to execute the procedures relevant to this state.
- Connected State: In this state the connection is open. The protocol activates the Connected State Protocol to execute the procedures relevant to this state.

Figure 6.2.1-1 provides an overview of the access terminal states and state transitions. All transitions are caused by indications returned from protocols activated by the Default Air-Link Management Protocol.

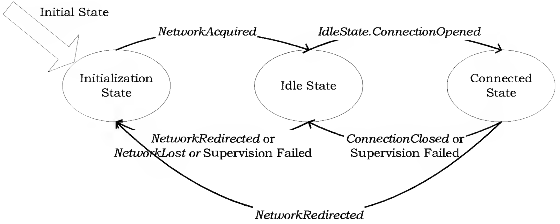

Initial State
NetworkAcquired
IdleState.ConnectionOpened
Initialization State
Idle State
Connected State
NetworkRedirected or NetworkLost or Supervision Failed
ConnectionClosed or Supervision Failed
NetworkRedirected

**Figure 6.2.1-1. Air Link Management Protocol State Diagram (Access Terminal)**

Figure 6.2.1-2 provides an overview of the access network states and state transitions.

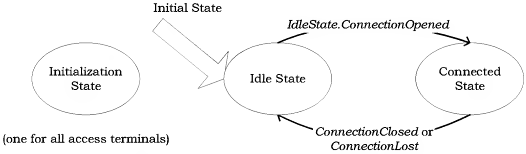

Initial State
IdleState.ConnectionOpened
Initialization State
Idle State
Connected State
(one for all access terminals)
ConnectionClosed or ConnectionLost

**Figure 6.2.1-2. Air Link Management Protocol State Diagram (Access Network)**

Table 6.2.1-1 provides a summary of the Connection Layer and MAC Layer protocols that are active in each state.

**Table 6.2.1-1. Active Protocols Per Air Link Management Protocol State**

| Initialization State | Idle State | Connected State |
|---|---|---|
| Overhead Messages Protocol | Overhead Messages Protocol | Overhead Messages Protocol |
| Initialization State Protocol | Idle State Protocol | Connected State Protocol |
| Control Channel MAC Protocol[7] | Route Update Protocol | Route Update Protocol |
| | Control Channel MAC Protocol | Control Channel MAC Protocol |
| | Access Channel MAC Protocol | Forward Traffic Channel MAC Protocol |
| | Forward Traffic Channel MAC Protocol[8] | Reverse Traffic Channel MAC Protocol |
| | Reverse Traffic Channel MAC Protocol[9] | |

### 6.2.2 Primitives and Public Data

#### 6.2.2.1 Commands

This protocol defines the following commands:

- *OpenConnection*
- *CloseConnection*

#### 6.2.2.2 Return Indications

This protocol does not return any indications.

#### 6.2.2.3 Public Data

- None.

### 6.2.3 Basic Protocol Numbers

The Type field for the Air-Link Management Protocol is one octet, set to $N_{ALMPType}$.

The Subtype field for the Default Air-Link Management Protocol is two octets, set to $N_{ALMPDefault}$.

[7] Activated by the Initialization State Protocol

[8] Only during connection setup

[9] Only during connection setup

### 6.2.4 Protocol Data Unit

The transmission unit of this protocol is a message. This is a control protocol; and, therefore, it does not carry payload on behalf of other layers or protocols.

This protocol uses the Signaling Application to transmit and receive messages.

### 6.2.5 Procedures

#### 6.2.5.1 Protocol Initialization and Configuration

This protocol shall be started in the Initialization State for the access terminal.

This protocol shall have a single instance operating in the Initialization State at the access network, serving all access terminals.

This protocol shall have a single instance for each access terminal with which the access network is currently maintaining a session. This instance shall be started in the Idle State.

This protocol does not have any initial configuration requirements.

#### 6.2.5.2 Command Processing

##### 6.2.5.2.1 OpenConnection

If the protocol receives the *OpenConnection* command in the Initialization State, the access terminal shall queue the command and execute it when the access terminal enters the Idle State.

The access network shall ignore the command in the Initialization State.

If the protocol receives this command in the Idle State:

- Access terminal shall issue an *IdleState.OpenConnection* command.
- Access network shall issue an *IdleState.OpenConnection* command.

If the protocol receives this command in the Connected State the command shall be ignored.

##### 6.2.5.2.2 CloseConnection

If the protocol receives the *CloseConnection* command in the Connected State:

- Access terminal shall issue a *ConnectedState.CloseConnection* command.
- Access network shall issue a *ConnectedState.CloseConnection* command.

If the protocol receives this command in any other state it shall be ignored.

#### 6.2.5.3 Initialization State

In the Initialization State the access terminal has no information about the serving access network. In this state the access terminal selects a serving access network and obtains time synchronization from the access network.

#### 6.2.5.3.1 Access Terminal Requirements

The access terminal shall enter the Initialization State when the Default Air-Link Management Protocol is instantiated. This may happen on events such as network redirection and initial power-on. A comprehensive list of events causing the Default Air-Link Management Protocol to enter the Initialization State is beyond the scope of this specification.

The access terminal shall issue an *InitializationState.Activate* command upon entering this state. If the access terminal entered this state because the protocol received a Redirect message and a Channel Record was received with the message, the access terminal shall provide the Channel Record with the command.

If the protocol receives an *InitializationState.NetworkAcquired* indication the access terminal shall issue an *InitializationState.Deactivate* command[10] and transition to the Idle State.

#### 6.2.5.3.2 Access Network Requirements

The access network shall constantly execute a single instance of the Initialization State Protocol. The access network shall constantly execute a single instance of the Overhead Messages Protocol.

### 6.2.5.4 Idle State

In this state the access terminal has acquired the access network but does not have an open connection with the access network.

#### 6.2.5.4.1 Access Terminal Requirements

##### 6.2.5.4.1.1 General Requirements

The access terminal shall issue the following commands upon entering this state:

- *IdleState.Activate*
- *RouteUpdate.Activate*
- *AccessChannelMAC.Activate.*

If the access terminal had a queued *OpenConnection* command, it shall issue an *IdleState.OpenConnection* command.

If the protocol receives an *IdleState.ConnectionOpened* indication, the access terminal shall perform the cleanup procedures defined in 6.2.5.4.1.2 and transition to the Connected State.

If the protocol receives a Redirect message, a *RouteUpdate.NetworkLost*, an *OverheadMessages.SupervisionFailed*, an *OverheadMessages.ANRedirected*, a

[10] Some of the *Deactivate* commands issued by this protocol are superfluous (because the commanded protocol already put itself in the Inactive State) but are specified here for completeness.

*ControlChannelMAC.SupervisionFailed*, an *AccessChannelMAC.SupervisionFailed*, or an *AccessChannelMAC.TransmissionFailure* indication, the access terminal shall:

- Issue a *RouteUpdate.Deactivate* command,
- Issue an *OverheadMessages.Deactivate* command,
- Issue a *ControlChannelMAC.Deactivate* command,
- Perform the cleanup procedures defined in 6.2.5.4.1.2, and
- Transition to the Initialization State.

#### 6.2.5.4.1.2 Idle State Cleanup Procedures

The access terminal shall issue the following commands when it exits this state:

- *IdleState.Deactivate*
- *AccessChannelMAC.Deactivate*

### 6.2.5.4.2 Access Network Requirements

#### 6.2.5.4.2.1 General Requirements

The access network shall issue the following commands upon entering this state:

- *IdleState.Activate*
- *RouteUpdate.Activate*

If the protocol receives an *IdleState.ConnectionOpened* indication, the access network shall perform the cleanup procedures defined in 6.2.5.4.2.2 and transition to the Connected State.

The access network may send the access terminal a Redirect message to redirect it from the current serving network and optionally, provide it with information directing it to another network. If the access network sends a Redirect message it shall

- Issue a *RouteUpdate.Deactivate* command,
- Perform the cleanup procedures defined in 6.2.5.4.2.2.

#### 6.2.5.4.2.2 Idle State Cleanup Procedures

The access network shall issue the following command when it exits this state:

- *IdleState.Deactivate*

### 6.2.5.5 Connected State

In the Connected State, the access terminal and the access network have an open connection.

#### 6.2.5.5.1 Access Terminal Requirements

##### 6.2.5.5.1.1 General Requirements

The access terminal shall issue the following command upon entering this state:

- *ConnectedState.Activate*

If the protocol receives a *ConnectedState.ConnectionClosed*, an *OverheadMessages.SupervisionFailed*, a *ControlChannelMAC.SupervisionFailed*, a *RouteUpdate.AssignmentRejected,* or a *ForwardTrafficChannelMAC.SupervisionFailed* indication, the access terminal shall:

- Issue a *RouteUpdate.Close* command,[11]
- Issue a *ControlChannelMAC.Deactivate* command,
- Issue an *OverheadMessages.Deactivate* command,
- Perform the cleanup procedure defined in 6.2.5.5.1.2,
- Transition to the Idle State.

If the protocol receives a Redirect message or an *OverheadMessages.ANRedirected* indication, the access terminal shall:

- Issue a *RouteUpdate.Close* command,[12]
- Issue a *ControlChannelMAC.Deactivate* command,
- Issue an *OverheadMessages.Deactivate* command,
- Perform the cleanup procedure defined in 6.2.5.5.1.2,
- Transition to the Initialization State.

##### 6.2.5.5.1.2 Connected State Cleanup Procedures

The access terminal shall issue the following command when it exits this state:

- *ConnectedState.Deactivate*

#### 6.2.5.5.2 Access Network Requirements

##### 6.2.5.5.2.1 General Requirements

The access network shall issue the following command upon entering this state:

- *ConnectedState.Activate*

[11] The Route Update Protocol takes care of closing the Forward Traffic Channel MAC and Reverse Traffic Channel MAC Protocols.

[12] The Route Update Protocol takes care of closing the Forward Traffic Channel MAC and Reverse Traffic Channel MAC Protocols.

If the protocol receives a *ConnectedState.ConnectionClosed*, or *RouteUpdate.ConnectionLost* indication, the access network shall:

- Issue a *RouteUpdate.Close* command,
- Perform the cleanup procedures defined in 6.2.5.5.2.2,
- Transition to the Idle State.

The access network may send the access terminal a Redirect message to redirect it from the current serving network and optionally, provide it with information directing it to another network. If the access network sends a Redirect message it shall:

- Issue a *RouteUpdate.Deactivate* command,
- Perform the cleanup procedures defined in 6.2.5.5.2.2,
- Transition to the Idle State.

#### 6.2.5.5.2.2 Connected State Cleanup Procedures

The access network shall issue the following command when it exits this state:

- *ConnectedState.Deactivate*

## 6.2.6 Message Formats

### 6.2.6.1 Redirect

The access network sends the Redirect message to redirect the access terminal(s) away from the current network; and, optionally, the access network provides it with information directing it to one of a set of different networks.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| NumChannel | 8 |

NumChannel instances of the following field

| | |
|---|---|
| Channel | 24 |

MessageID The access network shall set this field to 0x00.

NumChannel The access network shall set this field to the number of Channel records it is including in this message.

Channel This field shall be set to the channel that the access terminal should reacquire. The channel shall be specified using the standard Channel Record definition, see 10.1.

| | |
|---|---|
| **Channels** | CC FTC |
| **Addressing** | broadcast unicast |

| | |
|---|---|
| **SLP** | Best Effort |
| **Priority** | 40 |

### 6.2.7 Protocol Numeric Constants

| Constant | Meaning | Value |
|---|---|---|
| $N_{ALMPType}$ | Type field for this protocol | Table 2.3.6-1 |
| $N_{ALMPDefault}$ | Subtype field for this protocol | 0x0000 |

### 6.2.8 Interface to Other Protocols

#### 6.2.8.1 Commands Sent

This protocol issues the following commands:

- *InitializationState.Activate*
- *InitializationState.Deactivate*
- *IdleState.Activate*
- *IdleState.Deactivate*
- *IdleState.OpenConnection*
- *ConnectedState.Activate*
- *ConnectedState.Deactivate*
- *ConnectedState.CloseConnection*
- *RouteUpdate.Activate*
- *RouteUpdate.Deactivate*
- *RouteUpdate.Close*
- *OverheadMessages.Deactivate*
- *ControlChannelMAC.Deactivate*
- *AccessChannelMAC.Activate*
- *AccessChannelMAC.Deactivate*

#### 6.2.8.2 Indications

This protocol registers to receive the following indications:

- *InitializationState.NetworkAcquired*
- *IdleState.ConnectionOpened*
- *ConnectedState.ConnectionClosed*
- *RouteUpdate.ConnectionLost*

- *RouteUpdate.NetworkLost*
- *RouteUpdate.AssignmentRejected*
- *OverheadMessages.ANRedirected*
- *OverheadMessages.SupervisionFailed*
- *ControlChannelMAC.SupervisionFailed*
- *AccessChannelMAC.SupervisionFailed*
- *ForwardTrafficChannelMAC.SupervisionFailed*

## 6.3 Default Initialization State Protocol

### 6.3.1 Overview

The Default Initialization State Protocol provides the procedures and messages required for an access terminal to acquire a serving network.

At the access terminal, this protocol operates in one of the following four states:

- Inactive State: In this state the protocol waits for an *Activate* command.
- Network Determination State: In this state the access terminal chooses an access network on which to operate.
- Pilot Acquisition State: In this state the access terminal acquires a Forward Pilot Channel.
- Synchronization State: In this state the access terminal synchronizes to the Control Channel cycle, receives the Sync message, and synchronizes to system time.

Protocol states and events causing transition between states are shown in Figure 6.3.1-1.

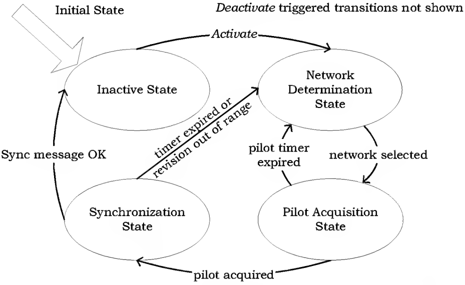


**Figure 6.3.1-1. Default Initialization State Protocol State Diagram**

### 6.3.2 Primitives and Public Data

#### 6.3.2.1 Commands

This protocol defines the following commands:

- *Activate* (an optional Channel Record can be specified with the command)
- *Deactivate*

#### 6.3.2.2 Return Indications

This protocol returns the following indications:

- *NetworkAcquired*

#### 6.3.2.3 Public Data

This protocol makes the following data public:

- Selected channel
- System time
- The following fields of the Sync message:
  - MaximumRevision
  - MinimumRevision
  - PilotPN

### 6.3.3 Basic Protocol Numbers

The Type field for the Initialization State Protocol is one octet, set to $N_{ISPType}$.

The Subtype field for the Default Initialization State Protocol is two octets, set to $N_{ISPDefault}$.

### 6.3.4 Protocol Data Unit

The transmission unit of this protocol is a message. This is a control protocol; and, therefore, it does not carry payload on behalf of other layers or protocols.

This protocol uses the Signaling Application to transmit and receive messages.

### 6.3.5 Procedures

The access network shall broadcast the Sync message periodically in a synchronous Control Channel capsule. This period should not exceed $T_{ISPSync}$ seconds.

The access network need not keep state for this protocol.

#### 6.3.5.1 Protocol Initialization and Configuration

This protocol shall be started in the Inactive State for the access terminal.

This protocol does not have any initial configuration requirements.

#### 6.3.5.2 Command Processing

The access network shall ignore all commands.

##### 6.3.5.2.1 Activate

If the protocol receives an *Activate* command in the Inactive State, the access terminal shall transition to the Network Determination State.

If the protocol receives this command in any other state, the access terminal shall ignore it.

#### 6.3.5.2.2 Deactivate

If the protocol receives a *Deactivate* command in the Inactive State, the access terminal shall ignore it.

If the protocol receives this command in any other state, the access terminal shall transition to the Inactive State.

### 6.3.5.3 Inactive State

In the Inactive State the access terminal waits for the protocol to receive an *Activate* command.

### 6.3.5.4 Network Determination State

In the Network Determination State the access terminal selects a CDMA Channel (see 10.1) on which to try and acquire the access network.

If a Channel Record was provided with the *Activate* command, the access terminal should select the system and channel specified by the record.

The specific mechanisms to provision the access terminal with a list of preferred networks and with the actual algorithm used for network selection are beyond the scope of this specification.

Upon selecting a CDMA Channel the access terminal shall enter the Pilot Acquisition State.

### 6.3.5.5 Pilot Acquisition State

In the Pilot Acquisition State the access terminal acquires the Forward Pilot Channel of the selected CDMA Channel.

Upon entering the Pilot Acquisition State, the access terminal shall tune to the selected CDMA Channel and shall search for the pilot. If the access terminal acquires the pilot, it shall enter the Synchronization State.[13] If the access terminal fails to acquire the pilot within $T_{ISPPilotAcq}$ seconds of entering the Pilot Acquisition State, it shall enter the Network Determination State.

### 6.3.5.6 Synchronization State

In the Synchronization State the access terminal completes timing synchronization.

Upon entering this state, the access terminal shall issue the *ControlChannelMAC.Activate* command.

If the access terminal fails to receive a Sync message within $T_{ISPSyncAcq}$ seconds of entering the Synchronization State, the access terminal shall issue a *ControlChannelMAC.Deactivate* command and shall enter the Network Determination State. While attempting to receive the

[13] The Access Terminal Minimum Performance Requirements contains specifications regarding pilot acquisition performance.

Sync message, the access terminal shall discard any other messages received on the Control Channel.

When the access terminal receives a Sync message:

- If the access terminal's revision number is not in the range defined by the MinimumRevision and MaximumRevision fields (inclusive) specified in the message, the access terminal shall issue a *ControlChannelMAC.Deactivate* command and enter the Network Determination State.
- Otherwise, the access terminal shall:
  - Set the access terminal time to the time specified in the message; The time specified in the message is the time applicable 160 ms following the beginning of the Control Channel Cycle in which the Sync message was received,
  - Return a *NetworkAcquired* indication,
  - Enter the Inactive State.

### 6.3.6 Message Formats

#### 6.3.6.1 Sync

The access network broadcasts the Sync message to convey basic network and timing information.

| Field | Length (bits) |
|---|---|
| MessageID | 2 |
| MaximumRevision | 8 |
| MinimumRevision | 8 |
| PilotPN | 9 |
| SystemTime | 37 |

MessageID The access network shall set this field to '00'.

MaximumRevision Maximum Air-Interface protocol revision supported by the access network. The access network shall set this field to the value specified in 1.14. This value shall be in the range [0x00, 0xff].

MinimumRevision Minimum Air-Interface protocol revision supported by the access network. The access network shall set this field to the value specified in 1.14. This value shall be in the range [0x00, MaximumRevision].

PilotPN Pilot PN Offset. The access network shall set this field to the pilot PN sequence offset for this sector in units of 64 PN Chips.

SystemTime The access network shall set this field to the System Time 160 ms after the start of the Control Channel Cycle in which this Sync message is being sent. The System Time is specified in units of 26.66... ms.

| | | | | |
|---|---|---|---|---|
| **Channels** | CCsyn | | **SLP** | Best Effort |
| **Addressing** | broadcast | | **Priority** | 30 |

### 6.3.7 Protocol Numeric Constants

| Constant | Meaning | Value | Comments |
|---|---|---|---|
| $N_{ISPType}$ | Type field for this protocol | Table 2.3.6-1 | |
| $N_{ISPDefault}$ | Subtype field for this protocol | 0x0000 | |
| $T_{ISPSync}$ | Sync message transmission period | 1.28 seconds | 3 × Control Channel Cycle |
| $T_{ISPPilotAcq}$ | Time to acquire pilot in access terminal | 60 seconds | |
| $T_{ISPSyncAcq}$ | Time to acquire Sync message in access terminal | 5 seconds | |

### 6.3.8 Interface to Other Protocols

#### 6.3.8.1 Commands Sent

This protocol issues the following commands:

- *ControlChannelMAC.Activate*
- *ControlChannelMAC.Deactivate*

#### 6.3.8.2 Indications

This protocol does not register to receive any indications.

## 6.4 Default Idle State Protocol

### 6.4.1 Overview

The Default Idle State Protocol provides the procedures and messages used by the access terminal and the access network when the access terminal has acquired a network and a connection is not open.

This protocol operates in one of the following four states:

- Inactive State: In this state the protocol waits for an *Activate* command.
- Sleep State: In this state the access terminal may shut down part of its subsystems to conserve power. The access terminal does not monitor the Forward Channel, and the access network is not allowed to transmit unicast packets to it.
- Monitor State: In this state the access terminal monitors the Control Channel, listens for Page messages and if necessary, updates the parameters received from the Overhead Messages Protocol. The access network may transmit unicast packets to the access terminal in this state.
- Connection Setup State: In this state the access terminal and the access network set-up a connection.

Protocol states and events causing the transition between the states are shown in Figure 6.4.1-1 and Figure 6.4.1-2.

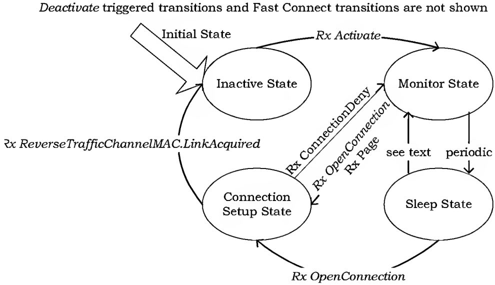


**Figure 6.4.1-1. Default Idle State Protocol State Diagram (Access Terminal)**

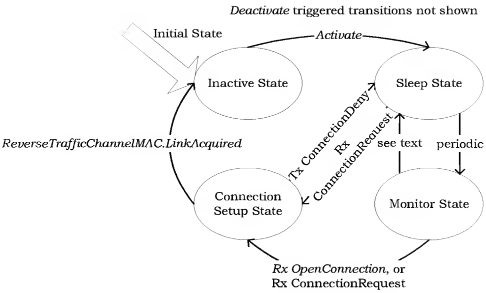


**Figure 6.4.1-2. Default Idle State Protocol State Diagram (Access Network)**

This protocol supports periodic network monitoring by the access terminal, allowing for significant power savings. The following access terminal operation modes are supported:

- Continuous operation, in which the access terminal continuously monitors the Control Channel.
- Suspended mode operation, in which the access terminal monitors the Control Channel continuously for a period of time and then proceeds to operate in the slotted mode. Suspended mode follows operation in the Air-Link Management Protocol Connected State and allows for quick network-initiated reconnection.
- Slotted mode operation, in which the access terminal monitors only selected slots.

This protocol supports two types of connection set-ups:

- Normal setup: this procedure is always performed at the initiative of the access terminal.[14] It consists of the access terminal sending a ConnectionRequest message which in turn causes the lower layers to open the connection. The Connection Setup State contains the requirements for normal setup.

[14] The access network may transmit a Page message to the access terminal directing it to initiate the procedure.

- Fast Connect: this procedure is always performed at the initiative of the access network and consists of the access network opening the connection directly via a *RouteUpdate.Open* command.[15] Fast Connect eliminates the need for the Page / ConnectionRequest exchange when the access network has pending data to transmit to an access terminal, and is especially useful when the access terminal is in suspended mode. Support for Fast Connect at the access network is optional. Support for Fast Connect at the access terminal is mandatory. The Monitor State contains the requirements for Fast Connect.

### 6.4.2 Primitives and Public Data

#### 6.4.2.1 Commands

This protocol defines the following commands:

- *Activate*
- *Deactivate*
- *OpenConnection*

#### 6.4.2.2 Return Indications

This protocol returns the following indications:

- *ConnectionOpened*
- *ConnectionFailed*

#### 6.4.2.3 Public Data

- None

### 6.4.3 Basic Protocol Numbers

The Type field for this protocol is one octet, set to $N_{IDPType}$.

The Subtype field for this protocol is two octets, set to $N_{IDPDefault}$.

### 6.4.4 Protocol Data Unit

The transmission unit of this protocol is a message. This is a control protocol; and, therefore, it does not carry payload on behalf of other layers or protocols.

This protocol uses the Signaling Application to transmit and receive messages.

[15] This command triggers a transmission of a TrafficChannelAssignment message based on the last RouteUpdate received from the access terminal.

### 6.4.5 Procedures

#### 6.4.5.1 Protocol Initialization and Configuration

This protocol shall be started in the Inactive State.

This protocol does not have any initial configuration requirements.

#### 6.4.5.2 Command Processing

##### 6.4.5.2.1 Activate

When the protocol receives an *Activate* command in the Inactive State:

- The access terminal shall transition to the Monitor State.
- The access network shall transition to the Sleep State.[16]

If the protocol receives this command in any other state it shall be ignored.

##### 6.4.5.2.2 Deactivate

When the protocol receives a *Deactivate* command in the Inactive State it shall be ignored.

When the protocol receives this command in any other state:

- The access terminal shall transition to the Inactive State.
- The access network shall transition to the Inactive State.

##### 6.4.5.2.3 OpenConnection

When the protocol receives an *OpenConnection* command in the Inactive State or the Connection Setup State, the command shall be ignored.

When the protocol receives this command in the Sleep State:

- The access terminal shall transition to the Connection Setup State.
- The access network shall queue the command and execute it when it is in the Monitor State.

When the protocol receives this command in the Monitor State:

- The access terminal shall transition to the Connection Setup State.
- The access network shall send a Page message to the access terminal and transition to the Connection Setup State.

#### 6.4.5.3 Inactive State

When the protocol is in the Inactive State it waits for an *Activate* command.

[16] Since the transitions happen asynchronously, this requirement guarantees that the access network will not transmit unicast packets to the access terminal over the Control Channel when the access terminal is not monitoring the channel.

- The access terminal should not monitor the Control Channel in this state.
- The access network shall not transmit unicast packets to the access terminal in this state.

### 6.4.5.4 Sleep State

When the access terminal is in the Sleep State it may stop monitoring the Control Channel by issuing the following commands:

- *OverheadMessages.Deactivate*
- *ControlChannelMAC.Deactivate*

The access terminal may shut down processing resources to reduce power consumption.

If the access terminal requires opening a connection, it shall transition to the Connection Setup State.

When the access network is in the Sleep State, it is prohibited from sending unicast packets to the access terminal.

If the access network receives a ConnectionRequest message, it shall transition to the Connection Setup State.

The access network and the access terminal shall transition from the Sleep State to the Monitor State in time to send and receive, respectively, the synchronous capsule sent in each Control Channel cycle $C$ satisfying

$$(C + R) \bmod N_{IDPSleep} = 0$$

where $C$ is the number of Control Channel cycles since the beginning of system time and $R$ is obtained as follows:

- If PreferredControlChannelCycleEnabled is equal to '0', then $R$ is the result of applying the hash function (see 10.4) using the following parameters:
  - Key = SessionSeed
  - Decorrelate = 6 × SessionSeed[11:0]
  - $N = N_{IDPSleep}$
  - where SessionSeed is given as public data of the Address Management Protocol.
- If PreferredControlChannelCycleEnabled is equal to '1', then $R$ is set to PreferredControlChannelCycle.

### 6.4.5.5 Monitor State

When the access terminal is in the Monitor State, it continuously monitors the Control Channel.

When the access network is in the Monitor State, it may send unicast packets to the access terminal.

6.4.5.5.1 Access Terminal Requirements

Upon entering the Monitor State, the access terminal shall issue the following commands:

- *OverheadMessages.Activate*
- *ControlChannelMAC.Activate*

The access terminal shall comply with the following requirements when in the Monitor State:

- Access terminal shall select the CDMA Channel as specified in 6.4.5.5.1.1.
- Access terminal shall monitor the overhead messages as specified in the Overhead Messages Protocol (see 6.8.5.5).
- If the access terminal receives a Page message, it shall transition to the Connection Setup State.
- If the access terminal requires opening a connection, it shall transition to the Connection Setup State.
- If the access terminal receives a *ReverseTrafficChannelMAC.LinkAcquired* indication it shall return a *ConnectionOpened* indication and transition to the Inactive State.[17]
- Access terminal may transition to the Sleep State if the requirements specified in 6.4.5.5.1.2 are satisfied.

6.4.5.5.1.1 CDMA Channel Selection

Each time the content of the SectorParameters message changes, the access terminal shall select a CDMA Channel from the list of channels in the message. If no channels are listed, the access terminal shall use the channel it is currently monitoring. If one or more channels are available, the access terminal shall use the hash function (see 10.4) to compute an index into the channel list provided in the message. The access terminal shall use the following hash function parameters to obtain this index:

- Key = SessionSeed
- Decorrelate = 0
- N = NumChannels field of the SectorParameters message

Where SessionSeed is provided as public data by the AddressManagement Protocol.

6.4.5.5.1.2 Transition to Sleep State

The access terminal may transition to the Sleep State if all of the following requirements are met:

- Access terminal has received at least one Control Channel synchronous capsule and has determined that the QuickConfig message and SectorParameters message are up to date (see 6.8.5.5).

[17] This requirement provides Fast Connect on the access terminal side.

- Access terminal received an *AccessChannelMAC.TxEnded* indication for every *AccessChannelMAC.TxStarted* indication it received since entering the Monitor State.[18]
- Access terminal has not advertised a suspend period that is current (see 6.5.5.3.1.1). The suspend period is current if the time advertised in the associated ConnectionClose message is greater than the current system time.[19]

### 6.4.5.5.2 Access Network Requirements

#### 6.4.5.5.2.1 General Requirements

- Access network shall select the CDMA Channel following the same specifications as the access terminal, see 6.4.5.5.1.1.
- If the access network requires opening a connection with the access terminal, it shall send it a Page message over the Control Channel.
- If the access network receives a ConnectionRequest message, it shall transition to the Connection Setup State.
- Access network may use an accelerated procedure to set-up a connection with the access terminal by bypassing the paging process. The access network should only use this procedure if it has a reasonable estimate of the access terminal's current location. To set-up a connection in an accelerated fashion (Fast Connect) the access network shall:
  - Issue a *RouteUpdate.Open* command.
  - Return a *ConnectionOpened* indication and transition to the Inactive State, if the protocol receives a *ReverseTrafficChannelMAC.LinkAcquired* indication.
- Access network shall transition to the Sleep State if the access terminal did not advertise a suspend period that is current.

### 6.4.5.6 Connection Setup State

The access terminal and the access network use the Connection Setup State to perform a normal connection set-up.

Figure 6.4.5.6-1 illustrates the process of opening a connection between the access terminal and the access network when this protocol is used along with the default Route Update and the default Reverse Traffic Channel MAC protocols.[20]

[18] This pairing ensures that the access terminal does not have any outstanding messages waiting for an answer.

[19] The access terminal monitors the Control Channel continuously during a suspend period thus avoiding the delay in opening access network initiated connections due to the sleep period.

[20] The Fast Connect message exchange is identical except for not having the Idle State Protocol ConnectionRequest message and the Route Update Protocol RouteUpdate message.

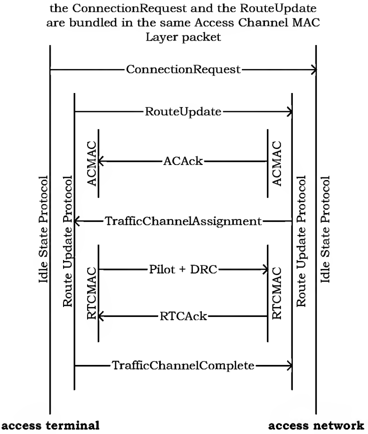


**Figure 6.4.5.6-1. Connection Setup Exchange**

6.4.5.6.1 Access Terminal Requirements

The access terminal shall comply with the following requirements.

- Upon entering the Connection Setup State the access terminal shall:
  - Issue an *OverheadMessages.Activate* command,
  - Issue a *ControlChannelMAC.Activate* command,
  - 
  - Send a ConnectionRequest message to the access network,
  - Set a state timer for $T_{IDPATSetup}$ seconds and start it after receiving an *AccessChannelMAC.TxEnded* indication,
- If the state timer expires, or if the access terminal receives a ConnectionDeny message, the access terminal shall issue a *RouteUpdate.Close* command, return a *ConnectionFailed* indication, and transition to the Monitor State,
- If the access terminal receives a *ReverseTrafficChannelMAC.LinkAcquired* indication, it shall return a *ConnectionOpened* indication and transition to the Inactive State.

#### 6.4.5.6.2 Access Network Requirements

If the access network denies the connection request, it should send the access terminal a ConnectionDeny message, shall return a *ConnectionFailed* indication, and shall transition to the Sleep State.

Otherwise, the access network shall perform the following:

- Set state timer for $T_{IDPANSetup}$ seconds.
- Issue a *RouteUpdate.Open* command.
- If the protocol receives a *ReverseTrafficChannelMAC.LinkAcquired* indication, the access network shall return a *ConnectionOpened* indication and transition to the Inactive State.
- If the state timer expires, the access network shall issue a *RouteUpdate.Close* command, return a *ConnectionFailed* indication, and transition to the Monitor State.

### 6.4.6 Message Formats

#### 6.4.6.1 Page

The access network sends the Page message to direct the access terminal to request a connection.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |

MessageID The access network shall set this field to 0x00.

| | | | |
|---|---|---|---|
| **Channels** | CC | **SLP** | Best Effort |
| **Addressing** | unicast | **Priority** | 20 |

#### 6.4.6.2 ConnectionRequest

The access terminal sends the ConnectionRequest message to request a connection.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| TransactionID | 8 |
| RequestReason | 4 |
| Reserved | 4 |

MessageID The access terminal shall set this field to 0x01.

TransactionID The access terminal shall increment this value for each new ConnectionRequest message sent.

RequestReason The access terminal shall set this field to one of the request reasons as shown in Table 6.4.6.2-1.

**Table 6.4.6.2-1. Encoding of the RequestReason Field**

| Field value | Description |
|---|---|
| 0x0 | Access Terminal Initiated |
| 0x1 | Access Network Initiated |
| All other values are invalid | |

Reserved The access terminal shall set this field to zero. The access network shall ignore this field.

| | | | | |
|---|---|---|---|---|
| **Channels** | AC | | **SLP** | Best Effort |
| **Addressing** | unicast | | **Priority** | 40 |

### 6.4.6.3 ConnectionDeny

The access network sends the ConnectionDeny message to deny a connection.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| TransactionID | 8 |
| DenyReason | 4 |
| Reserved | 4 |

MessageID The access network shall set this field to 0x02.

TransactionID The access network shall set this value to the TransactionID field of the corresponding ConnectionRequest message.

DenyReason The access network shall set this field to indicate the reason it is denying the connection, as shown in Table 6.4.6.3-1.

**Table 6.4.6.3-1. Encoding of the DenyReason Field**

| Field value | Description |
|---|---|
| 0x0 | General |
| 0x1 | Network Busy |
| 0x2 | Authentication or billing failure |
| All other values are reserved | |

Reserved The access network shall set this field to zero. The access terminal shall ignore this field.

| | | | | |
|---|---|---|---|---|
| **Channels** | CC | | **SLP** | Best Effort |
| **Addressing** | unicast | | **Priority** | 40 |

6.4.6.4 Configuration Messages

The Default Idle State Protocol uses the Generic Configuration Protocol for configuration. All configuration messages sent by this protocol shall have their Type field set to $N_{IDPType}$.

The following complex attribute and default values are defined (see 10.3 for attribute record definition):

| Field | Length (bits) | Default |
|---|---|---|
| Length | 8 | N/A |
| AttributeID | 8 | N/A |
| One or more of the following record: | | |
| ValueID | 8 | N/A |
| PreferredControlChannelCycleEnabled | 1 | '0' |
| PreferredControlChannelCycle | 0 or 15 | N/A |
| Reserved | 7 or 0 | N/A |

Length Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID The sender shall set this field to 0x00.

ValueID The sender shall set this field to an identifier assigned to this complex value.

PreferredControlChannelCycleEnabled

The sender shall set this field to '1' if PreferredControlChannelCycle field is included in this attribute; otherwise, the sender shall set this field to '0'.

PreferredControlChannelCycle

If PreferredControlChannelCycleEnabled is set to '1', the sender shall include this field and set it to specify the Control Channel Cycle in which the access terminal transitions out of the Sleep State (see 6.4.5.4) in order to monitor the Control Channel. The sender shall omit this field if PreferredControlChannelCycleEnabled is set to '0'.

Reserved The length of this field shall be such that the entire complex attribute is octet-aligned. The sender shall set this field to zero. The receiver shall ignore this field.

#### 6.4.6.4.1 ConfigurationRequest

The sender sends the ConfigurationRequest message to request the configuration of one or more parameters for this protocol. The ConfigurationRequest message format is given as part of the Generic Configuration Protocol (see 10.7).

The sender shall set the MessageID field of this message to 0x50.

| | | | |
|---|---|---|---|
| **Channels** | FTC RTC | **SLP** | Reliable |
| **Addressing** | unicast | **Priority** | 40 |

#### 6.4.6.4.2 ConfigurationResponse

The sender sends the ConfigurationResponse message to select one of the parameter settings offered in an associated ConfigurationRequest message. The ConfigurationResponse message format is given as part of the Generic Configuration Protocol (see 10.7).

The sender shall set the MessageID field of this message to 0x51.

| | | | |
|---|---|---|---|
| **Channels** | FTC RTC | SLP | Reliable |
| **Addressing** | unicast | Priority | 40 |

### 6.4.7 Protocol Numeric Constants

| Constant | Meaning | Value | Comments |
|---|---|---|---|
| $N_{IDPType}$ | Type field for this protocol | Table 2.3.6-1 | |
| $N_{IDPDefault}$ | Subtype field for this protocol | 0x0000 | |
| $N_{IDPSleep}$ | Number of control channel cycles constituting a sleep period | 0x0c | 5.12 seconds |
| $T_{IDPATSetup}$ | Maximum access terminal time in the Connection Setup State | 1.5 seconds | |
| $T_{IDPANSetup}$ | Maximum access network time in the Connection Setup State | 1 second | |

### 6.4.8 Interface to Other Protocols

#### 6.4.8.1 Commands Sent

This protocol issues the following commands:

- *RouteUpdate.Open* (access network only)
- *RouteUpdate.Close*
- *OverheadMessages.Activate*
- *OverheadMessages.Deactivate*
- *ControlChannelMAC.Activate*
- *ControlChannelMAC.Deactivate*

#### 6.4.8.2 Indications

This protocol registers to receive the following indications:

- *ReverseTrafficChannelMAC.LinkAcquired*
- *AccessChannelMAC.TxStarted*
- *AccessChannelMAC.TxEnded*

## 6.5 Default Connected State Protocol

### 6.5.1 Overview

The Default Connected State Protocol provides procedures and messages used by the access terminal and the access network while a connection is open.

This protocol can be in one of three states:

- Inactive State: In this state the protocol waits for an *Activate* command.
- Open State: In this state the access terminal can use the Reverse Traffic Channel and the access network can use the Forward Traffic Channel and Control Channel to send application traffic to each other.
- Close State: This state is associated only with the access network. In this state the access network waits for connection resources to be safely released.

Figure 6.5.1-1 and Figure 6.5.1-2 show the state transition diagrams at the access terminal and the access network respectively.

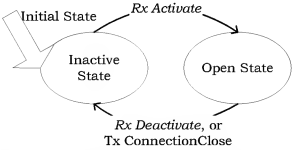


**Figure 6.5.1-1. Default Connected State Protocol State Diagram (Access Terminal)**

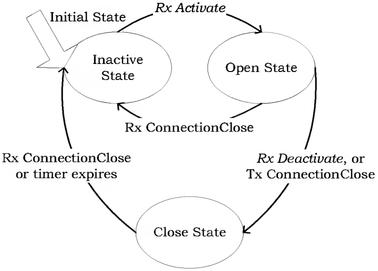


**Figure 6.5.1-2. Default Connected State Protocol State Diagram (Access Network)**

### 6.5.2 Primitives and Public Data

#### 6.5.2.1 Commands

This protocol defines the following commands:

- *Activate*
- *Deactivate*
- *CloseConnection*[21]

#### 6.5.2.2 Return Indications

This protocol returns the following indications:

- *ConnectionClosed*

#### 6.5.2.3 Public Data

- None

### 6.5.3 Basic Protocol Numbers

The Type field for the Connected State Protocol is one octet, set to $N_{CSPType}$.

The Subtype field for the Default Connected State Protocol is two octets, set to $N_{CSPDefault}$.

### 6.5.4 Protocol Data Unit

The transmission unit of this protocol is a message. This is a control protocol; and, therefore, it does not carry payload on behalf of other layers or protocols.

This protocol uses the Signaling Application to transmit and receive messages.

### 6.5.5 Procedures

#### 6.5.5.1 Protocol Initialization and Configuration

This protocol shall be started in the Inactive State.

This protocol does not have any initial configuration requirements.

#### 6.5.5.2 Command Processing

##### 6.5.5.2.1 Activate

When the protocol receives an *Activate* command in the Inactive State:

- The access terminal shall transition to the Open State.
- The access network shall transition to the Open State.

[21] The *CloseConnection* command performs the same function as the *Deactivate* command and is provided for clarity in the specification.

When the protocol receives this command in any other state it shall be ignored.

#### 6.5.5.2.2 Deactivate

When the protocol receives a *Deactivate* command in the Inactive State or in the Close State it shall be ignored.

When the protocol receives this command in the Open State:

- Access terminal shall send a ConnectionClose message to the access network and perform the cleanup procedures defined in 6.5.5.3.1.2.
- Access network shall send a ConnectionClose message to the access terminal, perform the cleanup procedures defined in 6.5.5.3.2.2, and transition to the Close State.

#### 6.5.5.2.3 CloseConnection

The access terminal and the access network shall process the *CloseConnection* command following the same procedures used for the *Deactivate* command, see 6.5.5.2.2.

### 6.5.5.3 Open State

In the Open State, the access terminal and the access network maintain a connection and can use it to exchange application traffic on the Reverse Traffic Channel, Forward Traffic Channel, and Control Channel.

#### 6.5.5.3.1 Access Terminal Requirements

#### 6.5.5.3.1.1 General Requirements

Upon entering the Open State, the access terminal shall issue the following commands:

- *OverheadMessages.Activate*
- *ControlChannelMAC.Activate*

The access terminal shall comply with the following requirements when in the Open State:

- The access terminal shall receive the Control Channel and the Forward Traffic Channel.
- The access terminal shall not transmit on the Access Channel.
- The access terminal shall monitor the overhead messages as specified in the Overhead Messages Protocol (see 6.8.5.5).
- If the access terminal receives a ConnectionClose message, it shall send a ConnectionClose message with CloseReason set to "Close Reply" and execute the cleanup procedures defined in 6.5.5.3.1.2.

If the access terminal sends a ConnectionClose message, it may advertise, as part of the ConnectionClose message, that it shall be monitoring the Control Channel continuously, until a certain time following the closure of the connection. This period is called a suspend period, and can be used by the access network to accelerate the process of sending a unicast packet (and specifically, a Page message or TrafficChannelAssignment message) to the access terminal.

#### 6.5.5.3.1.2 Cleanup Procedures

If the access terminal executes cleanup procedures it shall:

- Issue *RouteUpdate.Close* command.
- Return a *ConnectionClosed* indication.
- Transition to the Inactive State.

### 6.5.5.3.2 Access Network Requirements

#### 6.5.5.3.2.1 General Requirements

The access network shall comply with the following requirements when in the Open State:

- Access network shall receive the Reverse Traffic Channel and may transmit on the Forward Traffic Channel.
- If access network receives a ConnectionClose message, it shall consider the connection closed, and it should execute the cleanup procedures defined in 6.5.5.3.2.2 and transition to the Inactive State.
- If access network requires closing the connection, it shall transmit a ConnectionClose message, execute the cleanup procedures defined in 6.5.5.3.2.2, and transition to the Close State.

#### 6.5.5.3.2.2 Cleanup Procedures

When the access network performs cleanup procedures it shall:

- Issue *RouteUpdate.Close* command,
- Return a *ConnectionClosed* indication.

### 6.5.5.4 Close State

The Close State is associated only with the access network. In this state the access network waits for a replying ConnectionClose message from the access terminal or for an expiration of a timer.

Upon entering this state, the access network shall set a timer for $T_{CSPClose}$ seconds. If the access network receives a ConnectionClose message in this state, or if the timer expires, it may close all connection-related resources assigned to the access terminal, and should transition to the Inactive State.

## 6.5.6 Message Formats

### 6.5.6.1 ConnectionClose

The access terminal and the access network send the ConnectionClose message to close the connection.

| Field | Length (bits) |
|---|---|
| MessageID | 8 |
| CloseReason | 3 |
| SuspendEnable | 1 |
| SuspendTime | 0 or 36 |
| Reserved | variable |

MessageID The sender shall set this field to 0x00.

CloseReason The sender shall set this field to reflect the close reason, as shown in Table 6.5.6.1-1.

**Table 6.5.6.1-1. Encoding of the CloseReason Field**

| Field value | Description |
|---|---|
| '000' | Normal Close |
| '001' | Close Reply |
| '010' | Connection Error |
| All other values are reserved | |

SuspendEnable The access terminal shall set this field to '1' if it will enable a suspend period following the close of the connection. The access network shall set this field to '0'.

SuspendTime Suspend period end time. This field is included only if the SuspendEnable field is set to '1'. The access terminal shall set this field to the absolute system time of the end of its suspend period in units of 80 ms.

Reserved The length of this field shall be such that the entire message is octet-aligned. The sender shall set this field to zero. The receiver shall ignore this field.

| | | | |
|---|---|---|---|
| **Channels** | CC FTC RTC | **SLP** | Best Effort |
| **Addressing** | unicast | **Priority** | 40 |

### 6.5.7 Protocol Numeric Constants

| Constant | Meaning | Value | Comments |
|---|---|---|---|
| $N_{CSPType}$ | Type field for this protocol | Table 2.3.6-1 | |
| $N_{CSPDefault}$ | Subtype field for this protocol | 0x0000 | |
| $T_{CSPClose}$ | Access network timer waiting for a responding ConnectionClose message | 1.5 seconds | |

### 6.5.8 Interface to Other Protocols

#### 6.5.8.1 Commands Sent

This protocol sends the following commands:

- *RouteUpdate.Close*
- *OverheadMessages.Activate*
- *ControlChannelMAC.Activate*

#### 6.5.8.2 Indications

This protocol does not register to receive any indications.

## 6.6 Default Route Update Protocol

### 6.6.1 Overview

The Default Route Update Protocol provides the procedures and messages used by the access terminal and the access network to keep track of the access terminal's approximate location and to maintain the radio link as the access terminal moves between the coverage areas of different sectors.

This protocol can be in one of three states:

- Inactive State: In this state the protocol waits for an *Activate* command.
- Idle State: This state corresponds to the Air-Link Management Protocol Idle State. In this state, the access terminal autonomously maintains the Active Set. Route update messages from the access terminal to the access network are based on the distance between the access terminal's current serving sector and the serving sector at the time the access terminal last sent an update.
- Connected State: This state corresponds to the Air-Link Management Protocol Connected State. In this state the access network dictates the access terminal's Active Set. Route update messages from the access terminal to the access network are based on changing radio link conditions.

Transitions between states are driven by commands received from Connection Layer protocols and the transmission and reception of the TrafficChannelAssignment message.

The protocol states, messages and commands causing the transition between the states are shown in Figure 6.6.1-1.

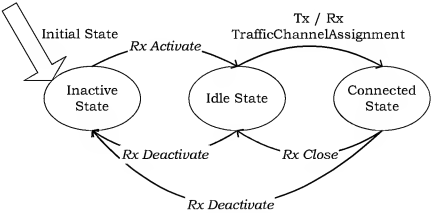


**Figure 6.6.1-1. Default Route Update Protocol State Diagram**

### 6.6.2 Primitives and Public Data

#### 6.6.2.1 Commands

This protocol defines the following commands:

- *Activate*
- *Deactivate*
- *Open*
- *Close*

#### 6.6.2.2 Return Indications

This protocol returns the following indications:

- *ConnectionLost* (access network only)
- *NetworkLost*
- *IdleHO*
- *ActiveSetUpdated*
- *RouteUpdate.AssignmentRejected*

#### 6.6.2.3 Public Data

This protocol shall make the following data public:

- Active Set
- Pilot PN for every pilot in the Active Set
- SofterHandoff for every pilot in the Active Set
- MACIndex for every pilot in the Active Set
- Channel record
- FrameOffset
- Current RouteUpdate message

### 6.6.3 Basic Protocol Numbers

The Type field for the Route Update Protocol is one octet, set to $N_{RUPType}$.

The Subtype field for the Default Route Update Protocol is two octets, set to $N_{RUPDefault}$.

### 6.6.4 Protocol Data Unit

The transmission unit of this protocol is a message. This is a control protocols; and, therefore, it does not carry payload on behalf of other layers or protocols.

This protocol uses the Signaling Application to transmit and receive messages.

### 6.6.5 Procedures

#### 6.6.5.1 Protocol Initialization and Configuration

This protocol shall be started in the Inactive State.

The access network may transmit a ConfigurationRequest message as part of the initial protocol configuration.

The access terminal shall be ready to receive a ConfigurationRequest message during initial protocol configuration.

This protocol shall use the Generic Configuration Protocol to process the ConfigurationRequest and ConfigurationResponse messages (see 10.7).

This protocol uses parameters that are provided, as public data by the Overhead Messages Protocol, or through ConfigurationRequest/ConfigurationResponse message exchanges, or by using a protocol constant. ConfigurationRequest and ConfigurationResponse messages can be sent initially as part of the session negotiation and in the Idle State and the Connected State.

Table 6.6.5.1-1 lists all of the protocol parameters obtained from the public data of the Overhead Messages Protocol. Section 6.6.6.5.1 lists the parameters that can be provisioned through a ConfigurationRequest message, along with the default values the access terminal shall use if it does not receive a ConfigurationRequest message. Section 6.6.7 lists the protocol constants.

**Table 6.6.5.1-1. Route Update Protocol Parameters that are Public Data of the Overhead Messages Protocol**

| RU Parameter | Comment |
|---|---|
| Latitude | Latitude of sector in units of 0.25 second |
| Longitude | Longitude of sector in units of 0.25 second |
| RouteUpdateRadius | Distance between the serving sector and the sector in which location was last reported which triggers a new report. If this field is set to zero, then distance triggered reporting is disabled |
| NumNeighbors | Number of neighbors specified in the message |
| NeighborPN | PN Offset of each neighbor in units of 64 PN chips |
| NeighborChannelIncluded | Set to '1' if a Channel Record is included for the neighbor |
| NeighborChannel | Neighbor Channel Record specifying network type and frequency |

### 6.6.5.2 Command Processing

#### 6.6.5.2.1 Activate

If the protocol receives an *Activate* command in the Inactive State, the access terminal and the access network shall transition to the Idle State.

If this command is received in any other state, it shall be ignored.

#### 6.6.5.2.2 Deactivate

If the protocol receives a *Deactivate* command in the Inactive State, it shall be ignored.

If the protocol receives this command in any other state, the access terminal and the access network shall:

- Issue a *ReverseTrafficChannelMAC.Deactivate* command,
- Issue a *ForwardTrafficChannelMAC.Deactivate* command,
- Transition to the Inactive State.

#### 6.6.5.2.3 Open

If the protocol receives an *Open* command in the Idle State,

- The access terminal shall ignore it.
- The access network shall:
  - Transmit a TrafficChannelAssignment message; the access network should base this message on the last RouteUpdate it received from the access terminal,
  - Issue a *ReverseTrafficChannelMAC.Activate* command,
  - Issue a *ForwardTrafficChannelMAC.Activate* command.
  - Transition to the Connected State.

If this command is received in any other state it shall be ignored.

#### 6.6.5.2.4 Close

If the protocol receives a *Close* command in the Connected State the access terminal and the access network shall:

- Issue a *ReverseTrafficChannelMAC.Deactivate* command,
- Issue a *ForwardTrafficChannelMAC.Deactivate* command,
- Transition to the Idle State.

If this command is received in any other state it shall be ignored.

### 6.6.5.3 Pilots and Pilot Sets

The access terminal estimates the strength of the Forward Channel transmitted by each sector in its neighborhood. This estimate is based on measuring the strength of the Forward Pilot Channel (specified by the pilot's PN offset and the pilot's CDMA Channel), henceforth referred to as the pilot.

When this protocol is in the Connected State, the access terminal uses pilot strengths to decide when to generate RouteUpdate messages.

When this protocol is in the Idle State, the access terminal uses pilot strengths to decide which sector's Control Channel it monitors.

The following pilot sets are defined to support the Route Update process:[22]

- Active Set: The set of pilots (specified by the pilot's PN offset and the pilot's CDMA Channel) associated with the sectors currently serving the access terminal. When a connection is open, a sector is considered to be serving an access terminal when there is a Forward Traffic Channel, Reverse Traffic Channel and Reverse Power Control Channel assigned to the access terminal. When a connection is not open, a sector is considered to be serving the access terminal when the access terminal is monitoring that sector's control channel.
- Candidate Set: The pilots (specified by the pilot's PN offset and the pilot's CDMA Channel) that are not in the Active Set, but are received by the access terminal with sufficient strength to indicate that the sectors transmitting them are good candidates for inclusion in the Active Set.
- Neighbor Set: The set of pilots (specified by the pilot's PN offset and the pilot's CDMA Channel) that are not in either one of the two previous sets, but are likely candidates for inclusion in the Active Set.
- Remaining Set: The set of all possible pilots (specified by the pilot's PN offset and the pilot's CDMA Channel) on the current channel assignment, excluding the pilots that are in any of the three previous sets.

At any given instant a pilot in the current CDMA Channel is a member of exactly one set.

The access terminal maintains all four sets. The access network maintains only the Active Set.

The access terminal complies with the following rules when searching for pilots, estimating the strength of a given pilot, and moving pilots between sets.

#### 6.6.5.3.1 Neighbor Set Search Window Parameters Update

The access terminal shall maintain RouteUpdateNeighborList which is a list of structures of type Neighbor (defined below). For each pilot (specified by the pilot's PN offset and the pilot's CDMA Channel) in the Neighbor Set, the access terminal shall maintain a structure in the RouteUpdateNeighborList.

A Neighbor structure consist of four fields: PilotPN, Channel, SearchWindowSize, and SearchWindowOffset.

The RouteUpdateNeighborList is used by the access terminal to perform pilot search on a pilot in the Neighbor Set.

When this set of procedures are invoked, the access terminal shall perform the following steps in the order specified:

- For each pilot (specified by its pilot PN and its channel) in the Neighbor Set, the access terminal shall first initialize the corresponding Neighbor structure in RouteUpdateNeighborList as follows:

[22] In this context, a pilot identifies a sector.

  - Set the structure's PilotPN field to the neighbor pilot's PN.
  - Set the structure's Channel field to the neighbor pilot's channel record.
  - Set the structure's SearchWindowSize field to the configurable attribute SearchWindowNeighbor.
  - Set the structure's SearchWindowOffset to zero.

- For each pilot (specified by the pilot's PN offset and the pilot's CDMA Channel) listed in the OverheadMessagesNeighborList, the access terminal shall set the non-NULL fields of the corresponding Neighbor structure in the RouteUpdateNeighborList to the fields of the Neighbor structure in the OverheadMessagesNeighborList for this pilot.
- For each pilot (specified by the pilot's PN offset and the pilot's CDMA Channel) listed in the NeighborListMessageNeighborList, the access terminal shall set the non-NULL fields of the corresponding Neighbor structure in the RouteUpdateNeighborList to the fields of the Neighbor structure in the NeighborListMessageNeighborList for this pilot.

6.6.5.3.2 Pilot Search

The access terminal shall continually search for pilots in the Connected State and whenever it is monitoring the Control Channel in the Idle State. The access terminal shall search for pilots in all pilot sets. This search shall be governed by the following rules:

1. <u>Search Priority</u>: The access terminal should use the same search priority for pilots in the Active Set and Candidate Set. In descending order of search rate, the access terminal shall search, most often, the pilots in the Active Set and Candidate Set, then shall search the pilots in the Neighbor Set, and lastly shall search the pilots in the Remaining Set.
2. <u>Search Window Size</u>: The access terminal shall use the search window size specified by the configurable attribute SearchWindowActive for pilots in the Active Set and Candidate Set. For each pilot in the Neighbor Set, the access terminal shall use the search window size specified by Table 6.6.6.5-1 and SearchWindowSize field of the corresponding Neighbor structure in the RouteUpdateNeighborList. The access terminal shall use search window size specified by configurable attribute SearchWindowRemaining for pilots in the Remaining Set.
3. <u>Search Window Center</u>: The access terminal should center the search window around the earliest usable multipath component for pilots in the Active Set. The access terminal should center the search window for each pilot in the Neighbor Set around the pilot's PN sequence offset plus the search window offset specified by Table 6.6.6.5-2 and SearchWindowOffset field of the corresponding Neighbor structure in the RouteUpdateNeighborList using timing defined by the access terminal's time reference (see 9.2.1.5). The access terminal should center the search window around the pilot's PN sequence offset using timing defined by the access terminal's time reference (see 9.2.1.5) for the Remaining Set.

### 6.6.5.3.3 Pilot Strength Measurement

The access terminal shall measure the strength of every pilot it searches. The strength estimate formed by the access terminal shall be computed as the sum of the ratios of received pilot energy per chip, $E_c$, to total received spectral density, $I_0$ (signal and noise) for at most *k* multipath components, where *k* is the maximum number of multipath components that can be demodulated simultaneously by the access terminal.

### 6.6.5.3.4 Pilot Drop Timer Maintenance

For each pilot, the access terminal shall maintain a pilot drop timer.

If DynamicThresholds is equal to '0', the access terminal shall start a pilot drop timer for each pilot in the Candidate Set or the Active Set whenever the strength becomes less than the value specified by PilotDrop. The access terminal shall set the timer value to expired after the time specified by PilotDropTimer. The timer shall be reset and disabled if, before it expires, the strength of the pilot becomes greater than the value specified by PilotDrop.

If DynamicThresholds is equal to '1', the access terminal shall perform the following:

- The access terminal shall start a pilot drop timer for each pilot in the Candidate Set whenever the strength of the pilot becomes less than the value specified by PilotDrop and the pilot drop timer shall be set to expired after the time specified by PilotDropTimer. The timer shall be reset and disabled if the strength of the pilot becomes greater than the value specified by PilotDrop before it expires.
- For each pilot in the Active Set, the access terminal shall sort pilots in the Active Set in order of increasing strengths, i.e., $PS_1 < PS_2 < PS_3 < ... < PS_{N_A}$, where $N_A$ is the number of the pilots in the Active Set. The access terminal shall start the timer whenever the strength $PS_i$ satisfies the following inequality:

$$10 \times \log_{10} PS_i < \max\left(\frac{\text{SoftSlope}}{8} \times 10 \times \log_{10} \sum_{j>i} PS_j + \frac{\text{DropIntercept}}{2}, -\frac{\text{PilotDrop}}{2}\right)$$

$$i = 1, 2, ..., N_A - 1$$

  The access terminal shall reset and disable the timer whenever the above inequality is not satisfied for the corresponding pilot.

Sections 6.6.5.3.6 and 6.6.5.6.3 specify the actions the access terminal takes when the pilot drop timer expires.

### 6.6.5.3.5 Active Set Management

The access terminal shall support a maximum Active Set size of $N_{RUPActive}$ pilots.

Rules for maintaining the Active Set are specific to each protocol state (see 6.6.5.5.1 and 6.6.5.6.1).

### 6.6.5.3.6 Candidate Set Management

The access terminal shall support a maximum Candidate Set size of $N_{RUPCandidate}$ pilots.